Karlsruhe Institute of Technology

Improving and Analysing Bingo Voting

zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften

von der Fakultät für Informatik
des Karlsruher Instituts für Technologie (KIT)

genehmigte

Dissertation

von

Christian Henrich

aus Marburg

Tag der mündlichen Prüfung:	5. Juli 2012
Erster Gutachter:	Prof. Dr. Jörn Müller-Quade
Zweiter Gutachter:	Juniorprof. Dr. Dennis Hofheinz

Contents

Acknowledgements

This work would never have been accomplished without the help and support of many people. I would like to express my gratitude to those who were personally involved in my work. But I also offer my thanks and respect to all the giants who laid the scientific foundations for this work.

First and foremost I would like to offer my gratitude to my Doktorvater Prof. Dr. Jörn Müller-Quade. He taught me the big picture and small details of cryptography and encouraged me to delve into this fascinating field. His offer to study under his guidance made this work possible and I am very grateful for his encouragement and support.

I also would like to thank Dr. Dennis Hofheinz for his support, his questions and especially his answers. I learned very much from him and it was a great honour as well as a pleasure to work with him.

My sincerest thanks go to my colleague Carmen Kempka, who played perhaps the biggest role in the emergence of this thesis. I am very thankful for the collaborations and mutual discussions. Without her this work would not be what it is.

As authors of the original Bingo Voting, Dr. Jens-Matthias Bohli Stefan Röhrich both have a big part in this work. I thank them for the joint work and also for their input for this thesis.

I offer my thanks to my colleagues, especially Daniel Kraschewski and Dirk Achenbach, for many discussions and their help with this work. They are also very much responsible for providing a genial and inspiring atmosphere which I believe was fundamental for our results.

For their tireless efforts to keep the institute running and often shielding me from organisatorial and technical concerns of the real world I would like to thank Carmen Manietta and Holger Hellmuth.

I would like to thank Dr. Aleks Essex and Dr. Jeremy Clark for their support during my stay in Canada as well as the exchange of ideas.

I would like to thank Prof. Dr. Urs Hengartner for inviting me to the University of Waterloo. I acknowledge the financial support of the the Karlsruhe House of Young Scientists (KHYS) for the research trip to Canada.

I am very grateful to my wonderful wife who had endured and supported me during my work on my thesis.

Last but not least I would like to thank my parents for supporting and guiding me throughout my life.

Abstract

Elections are a defining part of every democracy. By electing representatives, the people transfers its power to form a government. Consequently, the requirements for a democratic election are very high.

The German Federal Constitutional Court ruled in March 2008 that each voter must be able to verify the correctness of the election result without special skills. As a consequence of this ruling, the regulation allowing the use of voting machines for federal elections was deemed unconstitutional—the electoral principle of publicity was violated since the correctness of the result was no longer publicly verifiable. This made elections using hand counted paper ballots the only remaining way of conducting political elections in Germany.

Cryptographic voting schemes are evolving into a practical alternative to the traditional election using paper ballots. They offer not only correctness but individual verifiability. The challenge for a cryptographic voting scheme is to provide verifiability while protecting voter privacy. This is important in order to prevent vote-buying and coercion.

Bingo Voting is a cryptographic voting scheme proposed by Bohli, Müller-Quade and Röhrich in 2007. The original version provides verifiable correctness based on a trusted random number generator. Each voter receives a receipt that enables the voter to verify that the corresponding vote was counted correctly. But the receipt does not provide any information about how the voter voted to any third party.

This work presents the original version of Bingo Voting, describes experience gained during the first real-world election conducted with Bingo Voting, proposes several improvements to Bingo Voting addressing several shortcomings and weaknesses, and analyses the resulting voting scheme.

The experiences gained during this first election with Bingo Voting and the analysis presented in this work show that Bingo Voting is a practical alternative to the traditional election with paper ballots.

Zusammenfassung

Demokratische Wahlen

In einer demokratischen Wahl legitimiert das Volk als Souverän die Regierung. Eine demokratische Wahl muss daher sehr hohen Anforderungen genügen. Dies hat das Bundesverfassungsgericht in seiner Entscheidung vom 3. März 2009 erneut bestätigt. Mit dem Urteil wurde der Einsatz von Wahlmaschinen bei der Bundestagswahl 2005 für verfassungswidrig erklärt, da die Wähler die korrekte Ermittlung des Ergebnisses nicht überprüfen können. Seit dem Urteil wird in Deutschland nur noch das klassische Papierwahlverfahren für politische Wahlen eingesetzt.

Kryptographische Wahlverfahren

Es existieren Alternativen zur klassischen Papierwahl, die ebenfalls ein nachvollziehbar korrektes Ergebnis versprechen. Schon seit einigen Jahren werden verschiedene kryptographische Wahlverfahren vorgeschlagen, die den hohen Anforderungen an eine demokratische Wahl genügen und dabei zusätzliche wünschenswerte Eigenschaften bieten sollen.

Die wichtigste Eigenschaft für ein kryptographisches Wahlverfahren ist die *durchgehende Verifzierbarkeit* (end-to-end verifiability). Der Wähler bekommt die Möglichkeit, sowohl die korrekte Zählung der eigenen Stimme zu überprüfen als auch die korrekte Ermittlung des Gesamtergebnisses nachzuvollziehen. Im Gegensatz dazu ist bei einer Papierwahl die korrekte Zählung der eigenen Stimme nur mittelbar überprüfbar, indem der Wähler die Wahlurne nach Stimmabgabe genau beobachtet und die Auszählung komplett mitverfolgt. Gleichzeitig ist es für einen einzelnen Wähler unmöglich, mehr als ein Wahllokal durchgehend zu überprüfen.

Bingo Voting

Bingo Voting ist ein kryptographisches Wahlverfahren, das von Bohli, Müller-Quade und Röhrich an der Universität Karlsruhe (TH) entwickelt und 2007 erstmals vorgestellt wurde [BMQR07a, BMQR07b]. Das ursprüngliche Verfahren bildet die Grundlage für diese Arbeit und wird in Kapitel 3 vorgestellt.

Bingo Voting setzt zwar einen Rechner als Wahlmaschine ein, der Wähler muss dem Rechner aber nicht vertrauen. Er kann später selbst die Korrektheit des Wahlergebnisses nachprüfen. Dazu bekommt der Wähler einen Beleg, mit dem er die korrekte Zählung seiner Stimme prüfen kann. Die korrekte Ermittlung des Wahlergebnisses kann mit Hilfe öffentlicher Daten von jedem nachvollzogen werden, auch von Personen, die nicht an der Wahl teilgenommen haben. Die Korrektheit des Verfahrens beruht dabei nur auf einem vertrauenswürdigen Zufallszahlengenerator, der beispielsweise mechanisch realisiert werden kann. In der ursprüglichen Veröffentlichung wird ein Bingokäfig als Beispiel angeführt, der dem Verfahren auch den Namen gibt.

In seiner ursprünglichen Form besitzt Bingo Voting einige Schwächen. Gelangt ein Angreifer in den Besitz eines Belegs, kann er die zugehörige Stimme unbemerkt manipulieren, da der Beleg nicht mehr kontrolliert wird. Dieses Problem teilt Bingo Voting mit den meisten anderen kryptographischen Wahlverfahren. Zum Zeitpunkt der ersten Veröffentlichung von Bingo Voting war auch unklar, welche Auswirkungen die Bedienbarkeit eines Wahlverfahrens auf die Sicherheit und das Ergebnis der Wahl hat. Außerdem sind die öffentlichen Daten, die für die Verifikation des Wahlergebnisses veröffentlich werden müssen, sehr groß.

Praktische Erfahrungen mit Bingo Voting

Im Rahmen der Studierendenparlamentswahl der Universität Karlsruhe (TH) im Jahre 2008, bei der ein Prototyp von Bingo Voting eingesetzt wurde, konnten wertvolle Erfahrungen mit dem Verfahren gewonnen werden. Diese wurden in [BHMQ+08] veröffentlicht und werden in Kapitel 4 beschrieben.

Ein wichtiges Resultat ist, dass Wähler bei Bingo Voting erst nach dem Abgeben der Stimme mit dem Verfahren konfrontiert werden und dass der Verifikationsprozess optional ist. Es unterscheidet sich damit von Verfahren wie Punchscan, Scantegrity oder ThreeBallot, die spezielle Stimmzettel benötigen, und MarkPledge sowie dem Verfahren von Moran und Naor, bei denen der Wähler zufällige Zeichen in die Wahlmaschine eingeben muss, um den Beleg zu generieren.

Ein Nachteil von Bingo Voting, der sich ebenfalls in dieser Wahl gezeigt hat, ist die Tatsache, dass die Daten, die für die Verifikation veröffentlicht werden müssen, sehr groß sind.

Verbesserungen für Bingo Voting

Den wesentlichen Schwerpunkt dieser Arbeit bilden die Erweiterungen und Verbesserungen von Bingo Voting, die in Kapitel 5 beschrieben werden.

Für die Studierendenparlamentswahl war es nötig, Bingo Voting so zu erweitern, dass sich damit komplexe Wahlen durchführen lassen. Die dafür notwendigen Änderungen werden ausführlich dargestellt und diskutiert.

Das Problem, dass die öffentlichen Daten sehr groß wurden, wurde ebenfalls betrachtet. Die Arbeit stellt mehrere Veränderungen vor, die die Größe der öffentlichen Daten um etwa den Faktor 100 verringern. Dies ist ein wichtiger Schritt für die Praktikabilität des Verfahrens.

Für die Sicherheit von Bingo Voting ist es notwendig, dass die Zufallszahlen, die der vertrauenswürdige Zufallszahlengenerator während des Wahlvorgangs zieht, nicht vorher schon für eine Füllstimme gezogen wurden. Daher muss die Länge dieser Zufallszahlen so gewählt werden, dass die Wahrscheinlichkeit, dass eine Zufallszahl mehrfach gezogen wird, ausreichend gering ist. Da der Wähler die Korrektheit seines Belegs in der Wahlkabine überprüft, indem er die Zufallszahl auf dem Beleg mit der vom Zufallszahlengenerator angezeigten Zahl vergleicht, machen längere Zufallszahlen diesen Verifikationsschritt schwieriger. Diese Arbeit stellt eine Modifikation des Bingo-Voting-Protokolls vor, die die Wahrscheinlichkeit, dass eine mehrfach gezogene Zufallszahl zu einer unbemerkten Manipulation der Stimme führt, deutlich verringert. Diese Modifikation erlaubt es, kürzere Zufallszahlen zu verwenden, was den Vergleich der Zufallszahlen in der Wahlkabine erleichtert und damit Bedienbarkeit und Akzeptanz erhöht.

Diese Arbeit beschreibt außerdem einen Angriff, gegen den die meisten kryptographischen Wahlverfahren nicht ausreichend geschützt sind, da er eine Grun-

dannahme außer Kraft setzt. Für die Korrektheit der meisten Wahlverfahren ist es notwendig anzunehmen, dass ein Angreifer nicht weiß, welche Stimmen überprüft werden, und er damit bei jeder Manipulation Gefahr läuft, entdeckt zu werden. Gelangt der Angreifer hingegen in Besitz eines Belegs, kann er mit hoher Wahrscheinlichkeit ausschließen, dass die zugehörige Stimme überprüft wird. Diese Arbeit beschreibt eine Gegenmaßnahme gegen diesen Angriff, die sich leicht auf andere kryptographische Wahlverfahren übertragen lässt, die den Beleg für den Wähler mit einem Computer generieren.

Analyse und Evaluation

Ein weiterer Schwerpunkt dieser Arbeit ist die Evaluation der vorgeschlagenen Verbesserungen und die Analyse des resultierenden Verfahrens. Dies geschieht in Kapitel 6.

Diese Arbeit präsentiert eine ausführliche Liste der Annahmen, die für die Korrektheit und Sicherheit einer Wahl, die mit Bingo Voting durchgeführt wird, notwendig sind. Die Zusammenhänge zwischen den einzelnen Annahmen und den Sicherheitseigenschaften des Verfahren werden diskutiert.

Für die Evaluation wurden drei Szenarien basierend auf drei großen Wahlen der jüngsten Geschichte verwendet: die Bundestagswahl 2009 in Deutschland, die Präsidentschaftswahl 2008 in den Vereinigten Staaten von Amerika sowie die Parlamentswahl 2009 in Indien.

Die Arbeit gibt eine Formel an, mit der sich die Wahrscheinlichkeit, dass sich eine Stimme unbemerkt verändert werden kann, für eine Wahl in Abhängigkeit von der Anzahl der Wähler und der Länge der verwendeten Zufallszahlen berechnen lässt. Die Analyse ergibt, dass Zufallszahlen mit einer Länge von 30 bit ausreichen, um die erwartete Anzahl der Stimmen, die sich unbemerkt verändern lassen, auf unter 1 zu senken. Zufallszahlen mit 30 bit Länge lassen sich durch eine Zeichenkette mit sechs Zeichen darstellen, die für die leichtere Vergleichbarkeit in zwei Dreiergruppen gegliedert sind.

Die Arbeit gibt ebenfalls eine Formel an, mit der sich die Größe der Daten, die für die Verifikation veröffentlicht werden müssen, in Abhängigkeit von der Anzahl der Kandidaten und der Anzahl der Wähler berechnen lässt. Für die Parlamentswahl 2009 in Indien, die größte demokratischen Wahl in der Geschichte, wären die öffentlichen Daten 97,3 TB groß, wenn diese Wahl mit Bingo Voting durchgeführt worden wäre. Für die Bundestagswahl 2009 und die amerikanische Präsidentschaftswahl 2008 ergeben sich Größen von 723 GB beziehungsweise knapp 3 TB. Dabei stellen die Veränderungen am Verfahren im Vergleich zum Originalverfahren eine Verbesserung um den Faktor 100 dar.

Zusammenfassung und Ausblick

Diese Arbeit bringt Bingo Voting ein großes Stück näher an ein praktisches Wahlverfahren für politische Wahlen mit den höchsten Anforderungen. Die Sicherheit und Verifizierbarkeit, die Bingo Voting bietet, ist nicht geringer als die von herkömmlichen Wahlen mit Papierstimmzetteln.

Um Bingo Voting allerdings wirklich bei einer realen Wahl einzusetzen, müssen noch einige offene Fragen gelöst werden. Die Sicherheit des Verfahrens beruht darauf, dass der vertrauenswürdige Zufallszahlengenerator die Zahlen während des Wahlvorgangs echt zufällig wählt. Eine Realisierung dieses Zufallszahlengenerators, die für den Wähler nachvollziehbar ist und dennoch zuverlässig ausreichend

Zufall produziert, fehlt zur Zeit.

Bingo Voting ist zwar prinzipiell auch bei komplexen Wahlen einsetzbar, aber der Beleg wird mit der Anzahl der Kandidaten und der Anzahl der Stimmen, die jeder Wähler verteilen darf, größer. Dies macht vor allem die Überprüfung in der Wahlkabine bei Wahlen mit vielen Kandidaten und vielen Stimmen pro Wähler umständlich. Eine Untersuchung, bis zu welcher Komplexität eine Wahl mit Bingo Voting bedienbar ist, steht noch aus.

Die größte Unbekannte bei der Einschätzung, ob und wann Bingo Voting für politische Wahlen in Deutschland eingesetzt werden kann, ist die Interpretation des Urteils des Bundesverfassungsgerichts zum Einsatz von Wahlmaschinen. Bis jetzt gibt es keine neue, verfassungsgemäße Bundeswahlgeräteverordnung, die als Maßstab für jedes Wahlverfahren dient.

1. Introduction

Elections are the defining element of a democracy. In a democracy, from the Greek words δῆμος – people and κρατία – governance, the people as souvereign uses its power to form the government through an election. So it is not remarkable that political elections enjoy the highest importance and also the highest protection in a democracy and have almost ritual character. The role of elections in modern democracy is becoming more prominent and important with the trend of direct democracy or liquid democracy when elections are not only used to elect representatives but also to involve citizens in the decision making process.

Due to the high importance of elections, the requirements are very strict. Voters have to be sure that the result of an election is correct and represents the will of the people. The introduction of the concept of voter privacy, i. e. the fact that single votes must be kept secret, turned this requirement into a hard problem. This work presents and discusses some of the many suggestions and different approaches to solve this problem.

The traditional election with paper ballots uses an elaborate procedure to ensure correctness of the election and voter privacy at the same time. All important steps of the election are public, including the hand counting of the votes, allowing each voter to check the correctness of the tally. The only part that happens in private is when a voter enters the voting booth to mark the ballot. Many are convinced that the traditional election is the best implementation for a democratic election.

The introduction of voting machines to aid voters and poll workers in casting and tallying votes was received with mixed feelings as the complaint that lead to the ruling of the German Federal Constitutional Court (Bundesverfassungsgericht) has shown. Supporters of voting machines claim that the advantages of aided marking and tallying, namely the faster, less labour intensive and less error-prone tally and the support for disabled voters, outweighs the disadvantages. Opponents of voting machines claim that the introduction of mechanical or computerized devices into the voting process not only deters people from participating in an election due to fear of the unfamiliar devices but also claim that those devices make the process opaque for voters and therefore undemocratic. The decision of the German Federal Constitutional Court [off09] has clarified that the use of voting machines is not per se undemocratic, but has also set high standards for

the transparency of such devices.

Cryptographic voting schemes aim to provide a different approach to the problem of secure elections. Instead of trusting the procedure as with traditional paper based elections or trusting the manufacturer of a voting machine, cryptographic voting schemes use cryptographic protocols and proofs to produce a verifiably correct tally.

Bingo Voting is a cryptographic voting scheme originally developed by Bohli, Müller-Quade and Röhrich in 2007 [BMQR07a, BMQR07b]. Bingo Voting uses a voting machine to record the voter's choice but also issues a receipt to the voter. This receipt allows the voter to verify that their vote was counted correctly, but it does not contain information that tells a third party how the voter voted.

This work demonstrates that Bingo Voting is a practical alternative to the traditional election using paper ballots.

1.1 Contribution of this Work

This works presents several improvements for the original Bingo Voting scheme by Bohli, Müller-Quade and Röhrich [BMQR07a] and describes the experiences of a small election conducted with Bingo Voting.

The main contribution of this work is the presentation of several improvements and extensions for Bingo Voting. It gives a detailed description of the measures necessary to use Bingo Voting for more complex elections. This includes elections in which each voter may distribute several votes or larger elections that require the employment of more than a single voting machine. The improvements also significantly decrease the size of the public data required for the verification process and increase usability by allowing shorter random numbers to be used without a loss of security.

This work also includes a thorough analysis of the properties of Bingo Voting. It states the assumptions that are made and how they relate to different security properties. The analysis also includes a formula to predict the size of the public data for an election. It also gives the probability of a corrupted voting machine to change a vote without risking detection depending on the length of the random numbers used. This allows the voting authority to make an informed decision about the length of the random numbers required depending on the size of the election.

1.2 Structure of this Work

Chapter 2 gives a short introduction to elections in general, presents the terminology and notions used in this work, describes requirements for elections, introduces several cryptographic primitives and protocols that are necessary for many cryptographic voting schemes, presents security notions that are commonly used in the literature, and gives a description of a number of cryptographic voting schemes.

Chaper 3 presents the original version of Bingo Voting as described in the original publications [BMQR07a, BMQR07b].

Bingo Voting was employed in the student parliament elections of the University of Karlsruhe in 2008. Chapter 4 describes the election, the implementation of Bingo Voting that was used and the experiences that were gained during the election.

Chapter 5 describes several extensions and improvements for the original version of Bingo Voting. These improvements allow Bingo Voting to be employed in larger and more complex elections, reduce the size of the public data required for the verification process, increase usability by reducing the size of the random numbers that are used to encode the voter's choice on a receipt, enable the voter to dispute an election without unveiling the vote, and prevent the receipt stealing attack that poses a problem for almost all cryptographic voting schemes.

Chapter 6 presents an analysis of the Bingo Voting protocol and evaluates the improvements proposed in this work.

Chapter 7 summarizes and discusses the results and concludes this work.

2. Preliminaries

The topic of elections is very diverse and encompasses questions of politics, law, social sciences, and psychology as well as security and, in case of cryptographic voting schemes, cryptography.

This chapter discusses the necessary preliminaries in detail. Section 2.1 discusses elections in general. Section 2.2 defines the terminology and notions used in the remainder of this work. Section 2.3 gives an overview over the security notions and properties present in the literature. Section 2.4 presents cryptographic primitives that are used in the Bingo Voting scheme. Section 2.5 gives a short overview over cryptographic voting schemes in the literature.

2.1 About Elections

There are many different election types, systems, schemes and methods. All of these differ greatly in their functionalities and properties, how the tallies and results are determined, and how the voters experience the voting process and the whole election. The details greatly depend on the history and culture of the group of voters, the purpose of the election and on the means that are available.

This section gives an insight into the diversity of election and on the challenges that arise for a voting scheme. It also identifies the properties that most or all elections have in common.

2.1.1 Election Types

An election is essentially a way for a group of individuals (called voters) to reach a common decision by collecting the preferences of each individual on a clearly defined subject. This includes choosing one or more representatives, picking one of several alternatives, or accepting or declining a proposition.

Political elections are not the only occurrences of elections. The democratic principle has entered many aspects of the day-to-day live and culture of modern democracies. In this section we briefly discuss different types of elections.

In an election each voter expresses their preference on a set of choices or alternatives. The meaning of these choices and the preferences expressed depend of the election. As elections are used for many different purposes and on many occasions this varies a lot. In this section we use the term candidate to denote a

person that stands in an election. From the next section on we will use the term candidate more broadly (see Section 2.2).

The most prominent example for an election is the *political election* in which the people elect representatives and bestow them with legislative power. In this case, voters decide on political parties or candidates which best represent the voters' political will. Another example in which voters elect a person is to fill an office, often in judiciary or executive. In this case the choice is again between candidates but not for representative tasks. In a similar fashion, leading positions in most groups (non-political like clubs or associations as well as political parties or unions) are determined by an election. A peculiar case is the election of a board of directors in public companies, as here the number of votes a single shareholder possesses depends on their number of shares.

Another function of an election as a decision-making process. In this case, the voters do not decide on candidates but on actions, for example on a law proposition. This is an integral part of direct democracy as the citizens vote directly on policy initiatives. The results of these elections are binding for the government. Some forms of direct democracy exist in many democratic countries, especially for the case that the government is abusing its power the possibility of a motion of no-confidence ensures that the power remains with the people. A referendum is special form of an election in which the voters are asked to either accept or reject a particular proposal.

The oldest documented form of elections is the Athenian ostracism (from the Greek word ὄστρακον meaning pottery shard). In the annual procedure, the ancient Athenians chose one cititzen that was to be expelled from Athens for ten years. They used pottery shards to record their choice and the citizen with the most votes had to leave the city.

A weaker form of elections are opinion polls. Such polls are often held by private organisations for companies, media and the government. From a security point of view, these are very different from elections. As the result are in no way legally binding, it may be assumed that the entity conducting them is interested in the correctness of the result (as opposed to the voters themselves that are interested in the correctness of the result of an election). This is only partially true, since manipulated results of an opinion poll may also be used to sway the public opinion, e. g. by claiming that a certain opinion is less or more widespread than it actually is or by justifying political decisions.

The scope and impact of the election has the most influence on the security requirements. A non-binding opinion poll is often done without any security at all, a political election for the head of a state has very high security requirements.

For the remainder of this work we will only consider elections with high security requirements and completely abstract from the different possible subjects and purposes of an election.

2.1.2 Voting Procedure

The simplest form of a voting process is the public showing of hands. To vote on a certain subject, all eligible voters gather and raise their hand or use a similar public expression of their preference. One big advantage is that this process is trivially and obviously secure in the sense that every voter attending can tally and verify the result. One problem of this procedure is that it does not scale very

well and thus is unsuited for large groups (although technical aids can push this limit), especially since all voters have to be present at the same time. The second problem is that the voting process of each voter is public and everyone learns how a voter voted.

A public voting process may lead to influence on the voter and is nowadays uncommon for most political elections in which voters are citizens and not elected representatives. The secret ballot is a very important part of the modern voting culture. The origins of the modern secret ballot lies in the 18th and 19th century when several countries introduced secret ballots to reduce any undue influence on voters. Before the introduction of the secret ballot, vote buying and coercion were not uncommon. In most modern democracies this contradicts the requirement each voter should vote freely without being influenced.

In modern elections an often used medium to record votes is paper. It is common, easy to obtain and handle, the voter uses a pen to mark the ballot, and the tally is a manual counting process that often is public after the ballots are shuffled (and thus made anonymous) using a ballot box. The main disadvantage of the traditional paper ballot is that the voting process as well as the tally by hand takes time and is error-prone, especially when the election becomes more complex.

There have been several attempts to improve and accelerate the manual tally by introducing machines or computers to the voting process or the tally process. The least impact on the voting process itself is the use of scanners to electronically capture paper ballots marked by the voter by traditional means (i. e. a pen). The first approach to accelerate the tally, however, was made in America by introducing lever machines for casting and tallying the votes. Introducing devices directly into the voting process (and thus changing it) offers a wider spectrum of possibilities for improvements. A detailed discussion of the advantages and disadvantages of using paper ballots or machines for the voting process or the tally is in Section 2.1.6.

These are two inherently different situations a voter may be in when marking a ballot. The voter may either go to a polling station and mark the ballot in a designated area, called a voting booth, that ensures that the voter is unobserved during the voting process. Or the voter may fill the ballot at home and send the ballot to the place where it is tallied. The first scenario is called *presence voting* as the voter is present at a polling station when casting the vote. The second scenario is called *remote voting*. Section 2.1.7 discusses the two scenarios in detail.

2.1.3 Electoral Systems

After looking at the different subjects and purposes of an election and the different ways a vote may be cast, we now discuss how a voter may express their preferences. There are several ways of recording the preferences on a ballot and several ways of obtaining a result from the sum of all ballots cast. We call the exact method a *electoral system*, but it is sometimes also called a *voting system*.

The simplest electoral system is that the voter may choose one of the preferences and mark it, thus casting their vote for this preference. The choice with the most votes is declared winner of the election. This is often called a *winner-takes-all election* or *first-past-the-post voting*. There may be additional conditions, for example a quorum. A change of the German Basic Law, for example, is only successful if in the referendum at least two-thirds of the votes accept the proposal.

If there is more than one winner (for example when there are several seats to be won), the electoral system may either distribute the n seats to the n candidates with the most votes or distribute the seats according to the proportion of the votes. This is often case when voters do not elect individual candidates but groups of candidates, often parties. This system is called *proportional representation*.

In a different electoral system the voter has several votes instead of just one and may distribute those among the preferences. If voters are allowed to give more than one vote to a single preference we call this *cumulative voting*. This is often used for elections with proportional representation.

A different group of electoral system require the voter to not only cast one or several votes but instead rank the possible choices. When there is only one winner of the election, the resulting voting system is called *instant-runoff voting* or *alternative voting*. To determine the winner, the ballots are sorted by the candidates that were marked as first choice. If a single candidate has receive more than half of the votes, he wins the elections. If not, the candidate with the fewest votes is eliminated and the corresponding ballots are redistributed according to the candidates marked as second choice. This is repeated until a single candidate has received more that half of the votes cast in the election. In an election with more than one winner, the ranking of candidates is used by the so-called *single transferable vote* system. In this system, the ballots are tallied similarly to the instant-runoff system, but candidates are also eliminated when they received enough votes to win a seat and their ballots are redistributed accordingly.

All of these electoral systems may or may not allow for write-in candidates which allow the voter to add a candidate to the ballot and cast a vote for him.

2.1.4 Properties of Elections

Intuitively the requirements for an election are obvious and simple. Each voter should have the influence on the final result as specified by the voting system. This implies that only eligible voters may voter and that the final result correctly represents the will of the voters. It is also highly desireable that the final result is accepted by the participants. While the general principles are common consensus, details may vary from election to election and from country to country.

The properties elections have to satisfy are mostly deemed obvious or defined by regulations or laws. Depending on the importance and scale of the election the measures taken to ensure the security vary.

The rest of this section is specific for the situation in Germany if not noted otherwise. Most of these properties are true for other countries.

Article 38 of the Basic Law for the Federal Republic of Germany states the requirements for the election for the German Bundestag as follows:

> Members of the German Bundestag are to be elected in general, direct, free, equal and secret elections.

While this article only states the requirements for one specific election this has become the general standard for requirements for all political elections. We will briefly discuss the meaning of the requirements.

general Each citizen is allowed to cast a vote in the election.

direct The members of the Bundestag are elected directly by the voters.

free A voter may not be influenced by a third party when casting a vote.

equal All votes count equally.

secret The choice of a single voter must not become known to any third party.

Of these requirements stated by the German Basic Law, three are of special interest for voting schemes as discussed in this work. The requirements of free and secret elections mean that no information about the voter's choice may become known to any third party, as this may influence the voter. The requirement of equality of all votes includes the requirement that votes must not be omitted from the tally or altered by any third party.

The German Federal Constitutional Court has strengthened these requirements in a ruling in 2009 by calling the employment of voting machine in the Bundestag elections in 2005 unconstitutional. Section 6.4.3 discusses this decision and its impact on cryptographic voting schemes briefly.

These requirements are fundamental for elections and also the foundation for the formal definitions of security properties. Section 2.3 gives a detailed and more formal discussion of the security properties of cryptographic voting schemes.

2.1.5 Attacks on Voting Schemes

There are many ways to influence the results of an elections. Legal ways are, for example, election campaigns to influence voters. The fact that candidates and political parties spend large sums shows that there is an interest in a certain result of an election and a willingness to spend money for this. In addition to legal methods to influence an election, there are more direct and illegal ways to either persuade voters to vote in a certain way or to directly influence or change the result of the election.

Numerous examples have shown that elections pose the temptation of electoral fraud. Extreme cases are countries that are not truly democratic but try to use fraudulent elections to gain a semblance of democracy. But also truly democratic elections are threatened by attacks. This sections briefly presents several attacks on voting schemes. We will call the entity that attempts to manipulate an election an conducts the attacks the *adversary*.

Ballot Stuffing

A simple method of changing the result of an election is allowing voter to vote more than once. This is called *ballot stuffing*. In its simplest form this means that a voter casts more than one ballot, or votes at more than one polling station, but may also include impersonation of absentees.

Vote Buying

An adversary trying to change the result of an election may not only try to directly manipulate the result but instead convince voters to vote a certain way. If the voting scheme enables the adversary to check whether or not the voter behaved according to a previous arrangement, the adversary may use this to influence the voter by reward desired behaviour or penalize unwanted behaviour.

This attack is called *vote buying* or *coercion*. Breaking voter privacy automatically enables the adversary to buy votes and thus allows him to take undue

influence on the result of the elections. But it may also be possible for the adversary to be able to only break the secrecy of votes cast by coerced voters. An example for an election using paper ballots would be that coerced voters mark their ballots inconspicuously, e. g. by using ink that is only visible under UV light or by marking their choice using a + or χ instead of the normal $\times$.

Vote buying is considered to be the stronger attack as the voter is motivated to deviate from the voting protocol if this increases the chance of receiving payment.

A special form of an coercion attack is the *forced abstention* and the *forced randomization*. If the voter is able to recognize a randomized ballot, i. e. a ballot that was marked at random and not according to the preferences of the voter, or a ballot that denotes an abstention, the voter may be coerced to cast an abstention or a randomized ballot. This also allows the adversary to influence the election, even though the influence is subtle and those attacks are often disregarded.

Chain Voting

A voting scheme that uses paper ballots and hands each voter only a single ballot is often vulnerable to the following attack: The adversary obtains a single ballot, marks it according to his choice and hands it to a voter. The voter is now coerced (i. e. paid or threatened) to take the blank ballot the voter receives at the polling station, cast the filled ballot received from the adversary instead and bring back the blank ballot to the adversary. The adversary may now fill out the new blank ballot and coerce another voter to continue the attack. This attack is called *chain voting*.

The obvious countermeasure is allowing all voters free access to blank ballots. This is easy for simple paper ballots but may not be very practical for cryptographic voting schemes that use intricate (and expensive) ballots.

Psychological Aspects

For coercion attacks it is not necessary that the adversary is actually able to gain information about how the voter voted. It is sufficient that the voter believes that the adversary is able to detect if the voter deviates from the adversary's instructions.

The only way to face this problem seems to be voter education.

2.1.6 Paper vs. Machine

The introduction of mechanical and electronical devices into elections with the intention of making the voting process or the tallying process easier has sparked a debate of the benefits as well as costs and risks of these devices. The question whether elections should utilize computers for the voting process at all is disputed.

In this section we will briefly discuss the properties of paper based voting system as well as voting systems employing voting machines or computers. We will also consider devices only used to help in tallying the ballots but concentrate on the those that have an impact on the voting experience.

2.1.6.1 Paper Ballots

Paper ballots are a wide-spread, well-understood, well-accepted and user-friendly form to record votes. Paper ballots are easy and cheap to print, and a variety of pens may be used to mark a paper ballot. Every voter that is able to read and write is probably used to pen and paper and therefore has little problem marking a paper ballot.

The main disadvantage of paper ballots is that the tallying process is typically done by hand and as such long and error-prone. This is one of the main arguments for the use of voting devices.

The verification process for an election using paper ballot that are counted by hand is very straightforward. The auditor checks that the ballot box is empty at the beginning of the voting phase, and only eligible voter cast votes and each voter casts at most one vote during the voting phase. For the tally, the auditor verifies that the ballots are counted correctly. The problem of this verification process is that a single auditor is able to audit one polling station at most.

Another disadvantage is that paper ballots offer no feedback to the voter. In a complex election in which the voter may distribute a large number of votes, the voter may accidentally distribute too many or too few votes. A paper ballot does not give a warning when a voter is marking the ballot as invalid, for example by overvoting.

Paper ballots also are problematic for visually impaired and blind people. This may be solved using the Braille system to print the ballots.

2.1.6.2 Optical Scan Voting System

One major problem of elections using the traditional paper ballots is that the tally is time consuming and error-prone, especially for complex election. A seemingly simple way of alleviating this problem without any major changes to the voting process is the use of optical scan voting systems like scanners, digital pens or similar electronic devices that electronically record the voter's choice. These systems are common in the United States of America [Sal88]. In Germany, there was a single attempt to use digital pens in 2007 in Hamburg [AMBS07]. This plan was abandoned shortly before the election [Vol09].

The advantage of these devices is that the tallying process is much faster than counting by hand. However, the question of the security and reliability of these devices is fundamental to asses their usefulness. The security of the tallying process also greatly relies on the question whether these devices are used to obtain the final tally or to get a projection before tallying by hand. In the first case any security problems of the devices used jeopardize the integrity and correctness of the election. In the second case the biggest advantage of the devices is nullified. If the devices are monitored by recounting a small sample, there still remain open questions: If ballots are recounted, is the tally of the optical scan system or the result of the recount binding, how are these samples chosen, and under which conditions is a recount permitted or even required.

Optical scan voting systems seem to be a good compromise between paper ballots and voting devices. But without additional security measures they do not offer sufficient protection against manipulation. Saltman [Sal88] describes several cases in which computerized tallying may have lead to inaccurate results or even manipulations in elections.

2.1.6.3 Voting Machines

Direct recording electronic machines (DREs) offer an alternative way for voters to cast their votes. Instead of marking their choice on a paper ballot, a voter interacts with a computer that records the voter's choice. Tomz and Houweling offer a good overview over voting machines used in the USA in [TH03].

There are two major arguments that support the use of DREs. Since the voting machines record the ballots in digital form, the tally is very fast. The other argument is that that the voting machine interacts with the voter and is able to assist them during the voting process. Examples for this are the possibility of offering the voter a choice of different languages or assistance in complex elections by warning a voter when the ballot becomes invalid. They may also assist disabled voter, for example by offering an audio interface.

One big problem of voting machines is that the usability of the device depends on the familiarity of voters with computers in general. This may exclude especially elderly people from the election.

The biggest problem is, however, that without additional means of verifying the result the voting machine outputs at the end of the voting phase, voters and voting authority have no way of checking the correctness of the tally. In America, this has lead to the requirement that in addition to the electronic ballot there must be a physical copy of each voter's choice. The most common implementation is the voter verified paper audit trail (VVPAT) [GB07]. As for optical scan voting schemes, the question arises whether the physical copy or the result of the voting machine are binding and how test samples and recounts are handled.

After the employment of DREs for the Bundestag election 2005 in Germany, the German Federal Constitutional Court has found the use of voting machines unconstitutional if they do not allow every voter to verify the correctness of the election. See Section 6.4.3 for discussion of the decision of the German Federal Constitutional Court.

2.1.7 Presence vs. Remote Voting

In many elections, voters are not only able to cast their vote at a polling station but may alternatively mark their ballots at home and send them to the voting authority for the tally. This allows voters who are not able to visit a polling station during the voting phase to participate in the election. This is called *remote voting* in contrast to casting a vote in a polling station which is called *presence voting*. Traditionally the ballots are sent using the postal system. This variant of remote voting is called *postal voting*. Sending ballots in electronic form over the Internet offers another alternative called *Internet voting*. This section briefly discusses the differences of both presence voting and remote voting.

Presence voting means that voters cast their votes inside a polling station. They are required to enter a voting booth provided by the voting authority to mark the ballot. This offers the voter the ability to mark their ballot without any third party learning their choice which is an crucial prerequisite for the anonymity of the ballot and therefore of a free election.

When a voter uses remote voting, the voting authority does not know under which circumstances the ballot is marked or if it is even marked by the voter. Because of this, remote voting makes vote buying an coercion attacks simple. This weakness is accepted as remote voting allows voters to participate in the election even if they are unable or unwilling to cast their votes at a polling station. Remote voting also requires the voter to authenticate themselves when sending the ballot. The methods used differ greatly for postal voting and Internet voting and are also often vulnerable to attacks.

For the remainder of this work we will only consider presence voting schemes

which require the voter to enter a voting booth to mark the ballot or interact with a voting machine.

2.2 Terminology and Notions

For the more formal discussion of the properties of an election we require unambiguous notions. This section introduces the terminology and notions used in the remainder of this work.

2.2.1 Roles in an Election

In an election there are different roles that have to be fulfilled. This includes candidates, voters, poll workers and members of the Electoral Commission. To make analysis feasible we will abstract from several practical necessities and concentrate on the essential parts of an election.

We will consider an election as a protocol with a set of rules and a number of participants called *parties*. In the remainder of this work, the term 'party' is not used for a political party but unless otherwise noted stands for a participant in a protocol. We will treat an election as a protocol with three sets of parties: the set of *voting authorities*, the set of *voters*, and the set of *auditors*. A voting authority is a group of individuals entrusted with the organization of the election and related tasks. It is often convenient to treat the complete group as a single entity, but sometimes it is important to distinguish between several independent voting authorities and members with different tasks, for example poll workers. The set of voters encompasses all individuals that are eligible to vote in the election. The set of auditors consists of all individuals that observe the compliance with the electoral laws and verify the correctness of an election. In most cases every voter may also be an auditor, often even parties that are not voters may be in this group.

Elections are used for many purposes so the choices of the voter may be persons, political parties, approval for a suggestion or any other choice. In this work we will call the different choices *candidate* independently from whether they are individuals, political parties, different options or simply 'yes' and 'no'. While candidates play no role in the election protocol, individuals that are candidates may of course also be voters or auditors.

In addition to the legitimate roles, we have to consider an entity that attacks and attempts to disrupt or manipulate the election. We will call this entity the *adversary*. The adversary may *corrupt* any party participating in the election, taking complete control over their actions. The adversary may also *coerce* a voter. In this case the adversary does not take control of the actions of the voter but instead gives a set of instructions to the voter and receives a transcript of the voter's actions inside the polling station include any receipts the voter received. Intuitively, we will say that a cryptographic voting scheme satisfies a security property if no adversary that does not break the underlying assumptions is able to break the security property.

2.2.2 Phases of an Election

To describe the process of an election, we distinguish three phases that are strictly sequential. The *voting phase* is the middle phase during which the voters cast their votes. Before the voting phase the voting authority prepares the election.

After the voting phase, the voting authority tallies all votes and publishes the result.

Preparation Phase

Before voters cast their votes, the voting authority prepares everything that is required to conduct an election during the *preparation phase*. This includes the publication of the candidate list and any other information about the election that is required to be public, establishing the list of eligible voters (depending on the election this list may also be published), and organizing the poll workers and polling stations. It also includes anything specific to the voting scheme used. For the traditional election that uses paper ballot these must be printed and distributed, for most cryptographic voting schemes this means the publication of data that is later used for the verification of the result.

The preparation phase may start at an arbitrary point in time before the voting phase and lasts until the start of the voting phase. There may be practical and legal requirements for the length of time that must lay between certain preparations and the voting phase (e. g. the candidate list must be published well in advance so that voters are able to inform themselves).

Voting Phase

The *voting phase* is the phase in which each voter has the opportunity to cast a vote. From a theoretical point of view this phase lasts until every voter has either cast a vote or decided to not participate in the election. For practical reasons this phase lasts a predefined time, often one day or one week. It is assumed that all voters that choose to participate in the election have the opportunity to cast a vote within this predefined voting phase.

We call the process of the vote casting the *voting process* of a single voter. This encompasses everything a single voter does inside the polling station that is election specific, most prominently marking and casting a ballot or the interaction with a voting machine. This includes the poll workers verifying the eligibility of the voter, but since this procedure is depending more on the legal and practical requirements of the election than on the voting scheme used, this is often disregarded in the rest of this work.

Tally Phase

When all voters have cast their votes and the voting phase ends, the *tally phase* starts. During the tally phase, the votes are tallied and the data required for the verification process is published.

2.2.3 Tally and Result

This work will distinguish between the tally and the result of an election. For this work the tally is the sum of all votes cast by eligible voters during the voting phase. From this the result of the election is generated using a public and beforehand agreed upon algorithm that translates the sum of all ballots into the result of the election. This algorithm is part of the voting system but not part of the voting scheme.

For many elections this is simple if there is only one winner. When there are more than one winner, for example in an election in which a parliament with a number of seats is elected, this becomes more complicated. The tally algorithm

for the STV voting system mentioned in Section 2.1.3, for example, is rather sophisticated.

While the computation of the result and the algorithm that is used are important parts of an election it is mostly irrelevant for this work. The goal of all voting schemes discussed in this work is to collect and cumulate the votes of all voters and present a verifiably correct result. Nevertheless, the voting schemes have to support the voting system, especially voting systems that require ballots on which the candidates are ranked. For Bingo Voting this is non-trivial as shown in Section 5.1.3.

2.3 Security Notions

The requirements for a democratic elections are very high. Intuitively, these requirements are simple. The result of the election should be correct and no information about the choice of any voter should leak. For the assessment of cryptographic voting schemes a list of security properties and notions is used to denote the quality of a cryptographic voting scheme.

It is not trivial to decide whether a protocol is secure. One big problem is to specify what 'secure' means. The first notions of security were defined as being protected against specific attacks. A similar approach is defining security as a collection of specific security properties. The latest approach is simulation based security for protocols.

This section presents commonly used security properties and also discussed simulation based security definitions.

2.3.1 Correctness

It is essential that a cryptographic voting scheme outputs the correct tally which is the sum of all ballots cast. In addition, is also important that the auditors are able to verify that the tally is correct. If the output is not the correct tally, the verification process should indicate this.

It may be considered tolerable if the adversary changes a small number of votes. For many voting schemes it is impossible to guarantee that the tally is perfectly correct, since this often requires the assumption that every voter checks the correct inclusion of their vote (see Section 6.2.3). Also the counting of paper ballots by hand has always been error-prone. This is no problem as long as the error does not change the result of the election. These reasons make it tolerable that the adversary may be able to slightly change the tally, as long as the chance of being caught is large for large manipulations that change the result of the election.

It is important to note at this point that even the paper system makes no promises about the absolute correctness of the result as the counting process by hand produces errors. It is assumed that these counting errors are small and random. Supporters of traditional paper ballots correctly point out that paper ballots may be recounted to correct errors. However, this requires that the ballots are stored in a secure way to preclude manipulation between the first counting and the recount.

2.3.1.1 Software Independence

Rivest first defined the term *software independence* as in [Riv08] as follows:

> A voting system is *software-independent* if an undetected change or error in its software cannot cause an undete ctable change or error in an election outcome.

This notion is often considered to be implied by verifiable correctness or end-to-end security as defined in the next section. Software independence is a useful notion for voting schemes that use voting machines but no cryptography (such as voting machines with voter verified paper audit trails) as they often do not satisfy end-to-end security but still provide the possibility of detecting an error.

2.3.1.2 End-to-end Security

The requirement that a cryptographic voting scheme provides *verifiability* is an important part of the correctness. Most cryptographic voting schemes provide each voter with a receipt of their choice that allows the voter to verify that the corresponding ballot was correctly included in the final tally. This is called *individual verification*. The voting authority also publishes proofs that convince the auditors that the tally is indeed the congregation of all ballots cast. This is called *universal verification*. If the tally is not correct, at least one of the two verification steps should fail with high probability.

The notion that describes voting schemes that provide both individual and universal verification is *end-to-end security*, often abbreviated as E2E security or E2E verifiability. This term was originally used for voting schemes to describe an unbroken chain of custody for the ballots [Jon05] so that the voter was sure that the ballots that were tallied were the unaltered votes that were cast.

Küsters, Truderung and Vogt have schown in [KTV11] that individual and universal verification are not sufficient to guarantee the correctness of the tally if the universal verification does not include a proof that every ballot is well-formed. If only the voter is able to check that the ballot corresponding to their receipt is well-formed, a corrupted voter may be able to cast a ballot with negative votes as shown in [KTV11] on the example of ThreeBallot (cf. Section 2.5.7).

The notion of individual verifiability is often divided into two properties. The voter should be able to verify that the vote was *cast as intended*. In most cases this means convincing the voter that the receipt correctly encodes the voter's choice. This is typically done during the voting process inside the voting booth. Due to the additional requirement of receipt-freeness and coercion resistance this proof must not be transferable (see Section 2.3.2 for further details). After the voting phase, the voter should be able to verify that the ballot was *counted as cast*. This often requires the voter to check that a copy of their receipt was published by the voting authority.

2.3.2 Ballot Secrecy

Most electoral laws require a political election to be secret. The reason for this is that every voter should be able to decide freely, without risk of political persecution or undue influence from a third party. This requirement is called *voter privacy* or *ballot secrecy*.

From a theoretical point of view, a perfectly secure voting scheme without secrecy is trivial. Each voter marks a ballot with a unique id and receives a copy as receipt. After the voting phase, each ballot is published. This allows each voter

to verify that their ballots was published correctly and each auditor to reproduce the tally.

The requirement of ballot secrecy prevents this solution for modern elections. This makes the use of more elaborate voting schemes necessary that are able to provide both verifiable correctness and ballot secrecy.

The question whether correctness or ballot secrecy is more important has no easy answer. Correctness is surely very important in the short term, but privacy is more important in the long term. For an election, the correctness of the tally is important until the end of the tally phase and the announcement of the result. If the adversary is able to manipulate the election after the result was already implemented it is too late.

In [vdG09], van de Graaf gives two very compelling reasons for unconditional privacy:

> Already a decade ago it has been argued in the context of credential mechanism [Cha86] that privacy should be unconditional, since individuals cannot be expected to assess the strength of cryptographic mechanisms. In addition, since storage is becoming cheaper every day, we must assume that the data on the bulletin board will be stored forever. This means that the moment the cryptographic assumption on which the privacy of the ballots was based is broken, it will be possible to derive who voted for whom. In other words, with computational privacy we can almost be sure that 30 or 300 years from now we can know who voter for who. This could raise the possibility for some nasty scenarios, for instance a dictator who has come to power goes after people who have voted against him (or his father) several decades ago.

On the other hand, voter privacy is broken when the adversary observes the marking process or the interaction of the voter with the voting machine. This demonstrates that voter privacy is harder to ensure than correctness.

In the case of Bingo Voting, ballot secrecy is guaranteed under stronger assumptions than correctness, but if ballot secrecy is achieved it is unconditional and therefore everlasting. This is discussed in detail in Section 6.3.1.

2.3.2.1 Receipt Freeness

The term *receipt freeness* [BT94, Oka98, MN06] describes the fact that the voter receives no evidence that may be used to prove any third party how the voter voted. Cryptographic voting schemes often issue receipts for the individual verification process. This does not contradict the notion of receipt freeness if the receipt does not convince any third party of the voter's choice.

2.3.2.2 Coercion Resistance

An adversary that attempts to buy a vote or coerces a voter may request the voter to deviate from the voting process to obtain a receipt with additional information about the voter's choice. A voting scheme that does not give any information about the voter's choice even if the voter deviates from the intended voting process is called *coercion resistant*. This property was first defined in [JCJ05] as a notion that is strictly stronger than receipt freeness.

Küsters, Truderung and Vogt gave a game-based definition in [KTV09] and used it to prove that Bingo Voting is as coercion resistant as an ideal voting system which means that the voter is able to prove the participation in the election but nothing more.

Unruh and Müller-Quade proposed a general model of incoerbility [UMQ10] based on the UC framework by Canetti [Can00, Can01]. Unfortunately, the UC framework makes very strong statements about security of a protocol and currently no generally accepted formalism for a voting scheme exists in the UC framework.

2.3.3 Practical Requirements

In addition to the security requirements for correctness and privacy of an election, there are several practical requirements that are important or at least advantageous for a voting scheme.

2.3.3.1 Dispute Freeness

A verifiable election schemes guarantees that a voter is able to notice if their vote was not included correctly into the tally or if the tally is not correct. This is, however, often not sufficient. In addition to being able to notice manipulations, the voter should be able to convince the voting authority that the tally was indeed manipulated. On the other hand it should not be possible to claim that the tally is incorrect if this is not the case.

This property is often called *dispute freeness* as an voting scheme with this property allows a voter to prove a valid allegation. This means that if a suspicion is not proven it is invalid. Consequently, the voting authority is able to resolve all disputes.

An additional challenge is obtaining this property without threatening or weakening the voter's privacy. In many cases the voter notices if their vote was altered but has to unveil how they voted to prove the manipulation. A common suggestion to solve this problem is the use of trusted neutral arbiters that resolve the dispute but do not report the content of the disputed vote. While this is an unsatisfying solution as it requires the assumption that a sufficient number of such arbiters exists and that all arbiter are trustworthy to keep the content of disputed votes secret, this appears to be the only practical solution for many election schemes.

Section 5.4 proposes a dispute resolution procedure for Bingo Voting that allows the voter to dispute an election without divulging their choice.

2.3.3.2 Robustness

An election should not fail if a single corrupted voter deviates from the protocol or if a single member of the voting authority refuses to participate in the tallying process. A voting scheme that is able to tolerate a number of misbehaving parties is called *robust*. This property is important for real-world elections but hard to formalize.

2.3.4 Computational & Unconditional Security

One important kind of assumption are so-called computational assumptions. They say intuitively that it is not impossible but infeasible to solve a certain problem. Computational assumptions are quite common and very important for cryptography in general.

Encryption schemes are a prominent example for cryptographic schemes that rely on computational assumptions. In general, encryption schemes are only *computationally secure* as the adversary is always able to decrypt by guessing the correct key or just trying all keys until finding the correct one.

One notable exception the one-time pad which is an encryption scheme for which the security is not depending on a computational assumption. If the key is as long as the message and truly random, the adversary is unable to learn anything about a plaintext encrypted with a one-time pad except the length. This is independent of the computing power of the adversary and so we call schemes with this property *unconditionally secure*, *statistically secure* or *perfectly secure*.

For protocols we use the terms equivalently. The security property of a protocol is called *computational* if it relies on a computational assumption. If the property is independent of any assumptions, the protocol is said to have this property *unconditionally*.

Any system based on computational assumptions may be broken with sufficient computing power and enough time, or by new developments in algorithms or even computing paradigms. A protocol that is unconditionally secure is sometimes also said to offer *everlasting security* as it is impossible to break the security at any time in the future.

2.4 Cryptography

Most cryptographic voting schemes make heavy use of cryptography and cryptographic primitives. This section introduces the most common cryptographic primitives and concepts.

2.4.1 Bulletin Board

A bulletin board is a cryptographic building block used by many cryptographic voting schemes. It was formally described by Benaloh (né Cohen) and Fisher in [CF85], but the underlying principle is much older as actually even traditional voting systems rely on this primitive. The idea behind a bulletin board is that it is a secure asynchronous broadcast from one party, the voting authority, to all auditors. The bulletin board often includes a mechanism that automatically time-stamps each message that is posted. An additional property is that once something is posted onto the bulletin board it may not be erased or modified at a later point in time.

Traditional media like newspapers, radio and television have helped implementing this idea for elections by making all information about an election public. Today the implementation of this primitive is easier as the Internet has proven that once something of interest is published there it is very hard to remove it.

2.4.2 Commitments

A commitment scheme is a cryptographic two-party protocol with two phases. In the first phase, a sender commits to a value by sending a *commitment* to a receiver. At a later time the sender is able to open the commitment to the value by sending opening information to the receiver. This is called *opening*, *revealing* or *unveiling* the commitment. The receiver is able to verify that the commitment and the opening information, which also contains the value the sender was committed to, are consistent.

A commitment scheme has two important properties. The *binding* property ensures that the sender is able to open a commitment to only one value. This convinces the receiver that the value was fixed when the commitment was sent. The *hiding* properties guarantees that the receiver does not learn any information about the value contained in the commitment before receive the opening information.

If a commitment scheme is non-interactive, meaning that messages are only sent from the sender to the receiver of the commitment, it is easy to turn this commitment scheme from a two-party protocol into a protocol in which the receiver is the public (for example by using a bulletin board to publish all messages from the sender).

Bingo Voting uses unconditionally hiding discrete logarithm commitments (UHDLCs), often called Pedersen commitments. This commitment scheme is described in detail in the next section.

2.4.2.1 UHDLCs

Unconditionally hiding discrete logarithm commitments were first described by Chaum, Damgård and van de Graaf in [CDvdG87]. They are also called *Pedersen commitments* after the description by Pedersen in [Ped92]. The commitment scheme is unconditionally hiding. The binding property is based on the assumption that the discrete logarithm is hard to compute in a finite group (hence the name). The commitment scheme has the additional expedient property that it is rerandomizable.

A good candidate for G are the subgroup of quadratic residues of $\mathbb{Z}_p^\times$ with p being a safe prime, which means that $p = 2 \cdot q + 1$ and p, q prime. The implementation of Bingo Voting described in [BHMQ+08] used such a group with p approximately 1000 bits long. Another possibility is the use of an elliptic curve of prime order q as described in Section 5.5.1.

The public input of the sender and the receiver are a cyclic group $\mathbb{G}$ of order q in which the dlog-problem is hard, and two generators g, h of G chosen at random so that the discrete logarithm of g to the base h is unknown. To commit to a value $m \in \{1, \ldots, q\}$, the sender chooses a random value $r \in \{1, \ldots, q\}$, computes the commitment

$$c = g^m h^r$$

and sends c to the receiver. To open the commitment c, the sender simply sends the pair (m, r) and the receiver checks whether $c = g^m h^r$.

While UHDLCs are computationally binding, the binding property is reducible to the assumption that the discrete logarithm of g to base h, $\delta = \operatorname{dlog}_h g$, is unknown. Assume the sender is able to open a commitment c to two different contents, which means he is able to produce (m_1, r_1) and (m_2, r_2) with

$$c = g^{m_1} h^{r_1} = g^{m_2} h^{r_2} \tag{2.1}$$

and $m_1 \neq m_2$. With this unveil information the sender is able to express the discrete logarithm δ as

$$\operatorname{dlog}_h c = \delta \cdot m_1 + r_1 \tag{2.2}$$

$$\operatorname{dlog}_h c = \delta \cdot m_2 + r_2 \tag{2.3}$$

and from this he gets

$$\delta = \frac{r_1 - r_2}{m_1 - m_2}, \tag{2.4}$$

which contradicts the assumptions that δ is unknown to the sender.

The hiding property of UHDLCs is information theoretical as for any given commitment c and arbitrary m' there exists an r' such that $c = g^{m'}h^{r'}$. This means that even a computationally unbounded receiver is unable to determine what the content of a given commitment c is, unless m or r is revealed.

For Bingo Voting this property is essential to provide everlasting ballot secrecy.

2.4.2.2 Rerandomization

We say the commitment c to value m contains the randomness r if $c = g^m h^r$. With UHDLCs it is possible to change the randomness of a given commitment c and prove that both the old and the new, rerandomized commitment contain the same value m, even without knowing m.

To change the randomness, one multiplies the commitment c with $h^{\hat{r}}$ where $\hat{r}$ is fresh randomness. The new commitment

$$\hat{c} = c \cdot h^{\hat{r}} = g^m h^{r+\hat{r}} \tag{2.5}$$

now contains the randomness $r + \hat{r}$ if it contained the randomness r before. We call $\hat{c}$ the *rerandomization* of c and define the RERAND-function with

$$\textsc{Rerand}(c, \hat{r}) = c \cdot h^{\hat{r}} = g^m h^{r+\hat{r}} \tag{2.6}$$

for the rerandomization of c with the new randomness $\hat{r}$.

It is possible to convince any third party that two commitments c and $\hat{c}$ are commitments to the same value m by simply announcing the discrete logarithm $\hat{r}$ of $\frac{\hat{c}}{c}$ to the base h. If (m, r) opens c to m, then $(m, r + \hat{r})$ with

$$\hat{r} = \operatorname{dlog}_h \frac{\hat{c}}{c} = \operatorname{dlog}_h \frac{g^m h^{r+\hat{r}}}{g^m h^r} \tag{2.7}$$

opens $\hat{c}$ to the same value m.

The zero-knowledge proofs described in Section 2.4.4 use this special property of UHDLCs.

2.4.3 Zero-knowledge Proofs

An interactive proof system is a two-party protocol in which a prover $\mathcal{P}$ interacts with a verifier $\mathcal{V}$ in order to prove that a certain statement is correct.

Zero-knowledge proofs are interactive proof system with three properties which will be briefly described here.

Completeness If the statement is true, a verifier that adheres to the protocol is convinced and accepts the proof.

Soundness If the statement is false, there is no prover that is able to convince an honest verifier except with very small probability.

Zero-Knowledge For each verifier there exists a polynomially bounded simulator that is able to generate a transcript of an interaction of the verifier with an honest prover that is indistinguishable from a real interaction.

Intuitively, the zero-knowledge property means two things. First, no verifier learns anything during the interaction with an honest prover except that the statement is true, as any additional information could be used to distinguish between a real transcript and a simulated transcript. Second, the verifier is unable to convince any third party that the statement is true, as such a transcript could have been created by a simulator without interaction with the prover.

For a detailed introduction to zero-knowledge proofs see [Gol01].

2.4.3.1 Interactive Zero-knowledge Proofs

Zero-knowledge proofs are normally interactive and non-transferable. Since a transcript of a real interaction between a prover and a verifier is indistinguishable from a transcript created by a simulator that does not know whether the statement is true or not, such a transcript is not convincing to any third party.

This is an essential property for coercion resistance. A zero-knowledge proof is often used the following way for a coercion resistant voting scheme: The voter casts the vote and receives a zero-knowledge proof from the voting authority that his vote was recorded correctly. The zero-knowledge proof is not transferable and the voter is able to simulate a transcript of such a proof for any candidate. The coercer is unable to distinguish the real proof from the simulated ones and therefore learns nothing about which candidate the voter voted for. The voting scheme by Moran and Naor [MN06] uses zero-knowledge proofs this way to provide coercion resistance and verifiable correctness (see Section 2.5.6).

2.4.3.2 Non-interactive Zero-knowledge Proofs

In an election, the voting authority proves correctness of certain statements to the public. Zero-knowledge proofs are often used for their property that they leak no information other than the fact that the statement that is proven is true. Interactive zero-knowledge proofs are not practical for this as they would require each auditor to interact with the voting authority separately.

Fiat and Shamir proposed a method in [FS87] to turn an interactive zero-knowledge proof into a non-interactive zero knowledge proof. The idea is that the verifier does not choose the message he sends to the prover using true randomness but instead uses pseudorandomness generated by using a hash function with all messages received so far from the prover as input. The resulting proof is no longer depending on any random input of the verifier, so the prover is able to create the complete proof by himself. The proof is still convincing if the pseudorandomness used is large enough.

2.4.4 Proof of a Shuffle of Known Content

Bingo Voting uses zero-knowledge proofs at several points as parts of the proof of the correctness of the tally while protecting voter privacy. Section 3.4 describes in detail at which points of the verification process the proofs are employed. All of these proofs are proofs of a shuffle of known content.

A prover $\mathcal{P}$ proves to a verifier $\mathcal{V}$ that a certain set of commitments contains a certain list of plaintexts without showing which commitment has which content. For Bingo Voting it is important that the verifier is able to check that if a plaintexts exists n times on the list there are exactly n commitments with this plaintext as content.

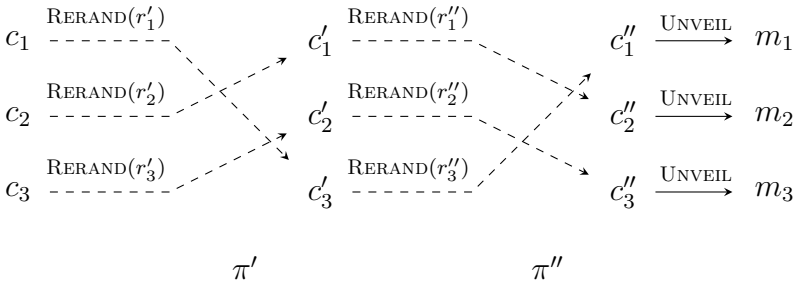


Figure 2.1: An example of a zero-knowledge proof using randomized partial checking showing that the set of commitments $C = \{c_1, c_2, c_3\}$ contains the plaintexts $M = (m_1, m_2, m_3)$. The public input of prover $\mathcal{P}$ and verifier $\mathcal{V}$ is C and M. $\mathcal{P}$ generates $c'_1, c'_2, c'_3, c''_1, c''_2, c''_3$ and sends it to $\mathcal{V}$ as well as the unveil information for c''_i (which is $m_i, r_{\pi'^{-1}(i)} + r'_{\pi'^{-1}(i)} + r''_{\pi''^{-1}(i)}$) for $i = 1, 2, 3$. $\mathcal{V}$ chooses a bit b and sends it to $\mathcal{P}$. If $b = 0$, $\mathcal{P}$ sends π' and r'_1, r'_2, r'_3, otherwise $\mathcal{P}$ sends π' and r''_1, r''_2, r''_3.

The original version of Bingo Voting uses randomized partial checking, the improved version uses a proof style proposed by Groth in [Gro02, Gro10]. Both proofs are described in this section.

2.4.4.1 Randomized Partial Checking

The public input of $\mathcal{P}$ and $\mathcal{V}$ is a set C of commitments and a list M of plaintexts in addition to the group $\mathbb{G}$ and the two generators g and h used for the commitment scheme. A single plaintext may be contained in M more than once. $\mathcal{P}$ wants to convince $\mathcal{V}$ that for each plaintext $m_i \in M$ there exists exactly one commitment $c_j \in C$ such that that m_i is the content of c_j. One simple proof concept for this is the *randomized partial checking* (RPC) that was used in the original Bingo Voting scheme and is explained in this section.

Randomized partial checking makes use of the rerandomization property of UHDLCs (see Section 2.4.2.2). The public input of prover and verifier is the set of commitments $C = \{c_1, \ldots, c_n\}$ and the list of plaintexts $M = (m_1, \ldots, m_n)$. The secret input of the prover is the set of random values $R = \{r_1, \ldots, r_n\}$ and a permutation π such that

$$(m_{\pi(i)}, r_{\pi(i)}) \qquad \text{opens} \qquad c_i \tag{2.8}$$

which is equivalent to saying that $m_{\pi(i)}$ is the content of commitment c_i with randomness $r_{\pi(i)}$.

The prover now chooses two new random permutations π' and π'' such that $\pi'' \circ \pi' = \pi$, and two sets of random values $R' = \{r'_1, \ldots, r'_n\}$ and $R'' = \{r''_1, \ldots, r''_n\}$. With this $\mathcal{P}$ computes

$$C' = \{c'_1, \ldots, c'_n\} \qquad \text{where} \qquad c'_{\pi'(i)} = \text{RERAND}(c_i, r'_i) \tag{2.9}$$

and

$$C'' = \{c''_1, \ldots, c''_n\} \qquad \text{where} \qquad c''_{\pi''(i)} = \text{RERAND}(c'_i, r''_i) \tag{2.10}$$

and sends these two sets of rerandomized commitments to $\mathcal{V}$.

The verifier now chooses a bit b at random and sends it back to the prover as challenge. If $b = 0$, the prover sends π' and R' back to the verifier, otherwise the prover sends π'' and R''. The verifier now checks whether Equation 2.9 or Equation 2.10 holds (depending on the choice of b).

It is easy to see that a cheating prover is caught with probability $\frac{1}{2}$. If the proof is repeated k times, the level of confidence is $(1 - 2^{-k})$.

A proof using randomized partial checking is an interactive zero-knowledge proof. To turn this into a non-interactive proof, the original paper suggests using the Fiat-Shamir method [FS87] (see Section 2.4.3). The main disadvantage of this type of proof is the size of the public data. For a confidence of $(1-2^{-80})$ the proof has to be repeated 80 times. Section 5.5 proposes a different proof system that results in much smaller proofs. Section 6.4.2 gives a comparison of the size of the public data with RPC style proofs and with the new proof system.

2.4.4.2 Efficient Proof of a Shuffle of Known Content by Groth

Groth has proposed a more efficient proof of a shuffle of known content which helps reducing the size of the public data of an election employing Bingo Voting as described in Section 5.5.2. The proof described by Groth in [Gro02, Gro10] is a generalized proof for homomorphic encryption schemes and is described here for UHDLCs.

The resulting proof technique is an interactive protocol between prover $\mathcal{P}$ and verifier $\mathcal{V}$. Both have a set of commitments C and a list of plaintexts M as common input. $\mathcal{P}$ will prove that for each commitment $c_i \in C$ there is one plaintext $m_j \in M$ that is the content of c_i, and that for each plaintext $m_s \in M$ there is a commitment $c_t \in C$ for which it is the content of. This is done without $\mathcal{V}$ learning anything more than this fact, in particular not which plaintext is the content of which commitment.

This protocol utilizes the fact that the message space $\mathbb{Z}_q$ is an integral domain and that a polynomial of degree n has at most n roots. The intuition behind the protocol is that $\mathcal{P}$ proves that the polynomial

$$P(X) = \prod_{i=1}^{n} m_i - X \tag{2.11}$$

with $m_1, \ldots, m_n \in M$ is identical to a polynomial Q derived from the commitments so that is contains the same roots m_i but in permuted order if and only if C is a shuffle of the content of M. To prove that both polynomials are identical, $\mathcal{V}$ chooses a point of evaluation x and then $\mathcal{P}$ uses a three-move multiplication proof of knowledge [DJ01] that the content of

$$\frac{c_1}{\textsc{Commit}\,(x)}, \ldots, \frac{c_1}{\textsc{Commit}\,(x)} \tag{2.12}$$

is the same as

$$\prod_{i=1}^{n} m_i - x. \tag{2.13}$$

Protocol

This protocol provides a proof of a shuffle of known contents as described by Groth in [Gro02].

Common input commitments $c_1, \ldots, c_n \in \mathbb{G}$ and plaintexts $m_1, \ldots, m_n \in \mathbb{Z}_q$

Secret input of $\mathcal{P}$ permutation π and randomness $r_1, \ldots, r_n$ so that $c_i = \textsc{Commit}\left(m_{\pi(i)}, r_i\right)$ for all $i = 1, \ldots, n$.

Initial Challenge $\mathcal{P}$ receives $x \in \mathbb{Z}_q$ chosen at random by $\mathcal{V}$, both set $\tilde{c}_i := c_i \cdot g^{-x}$

Multiplication Proof

1. $\mathcal{P}$ sets $p_1 := m_{\pi(1)} - x$ and for $i = 1, \ldots, n$ calculates $p_i := \prod_{j=1}^{i}\left(m_{\pi(j)} - x\right)$ and $c_{p_i} := \textsc{Commit}\left(p_i, r_{p_i}\right)$ with r_{p_i} chosen at random except for r_{p_n} which is set to 0 (so $c_{p_n} := \textsc{Commit}\left(p_n, 0\right)$).
2. For each $i = 2, \ldots, n$ $\mathcal{P}$ chooses a_i at random and sends c_{p_i}, $c_{a_i} := \textsc{Commit}\left(a_i, r_{a_i}\right)$ and $c_{b_i} := \textsc{Commit}\left(\left(m_{\pi(i)} - x\right) \cdot a_i, r_{b_i}\right)$.
3. $\mathcal{V}$ sends y_i chosen at random.
4. $\mathcal{P}$ sends $s_i := p_{i-1} \cdot y_i + a_i$, $r_{s_i} := r_{p_{i-1}} \cdot y_i + r_{a_i}$ and $d_i := r_i\left(p_{i-1} \cdot y_i + a_i\right) - r_{p_i} \cdot y_i - r_{b_i}$.
5. $\mathcal{V}$ checks whether
$$\frac{(\tilde{c}_i)^{s_i}}{(c_{p_i})^{y_i} \cdot c_{b_i}} = h^{d_i}$$
and whether $c_{p_n} = g^{\prod (m_i - x)}$.

2.5 Cryptographic Voting Schemes

The methods for conducting elections and tallying the voters' choices has changed over the long history of elections. The traditional election using paper ballots and ballot boxes has become the most used method since voter privacy was made mandatory for most elections.

The goal of most cryptographic voting schemes is to provide voters and auditors with the means to verify the tally of an election. There are many approaches to provide the necessary information without impairing voter privacy. They greatly differ in terms of the technology they use and their properties.

In addition to cryptographic voting schemes that require the voter to vote at a polling station there are also many schemes for secure voting over the internet. These schemes may use similar technologies and methods, and aim for the same goal, namely verifiably correct elections with voter privacy, but the situation is different. The differences between presence voting and remote voting are discussed in Section 1, in this chapter we will only consider cryptographic voting schemes for presence voting.

2.5.1 Farnel and Twin

Most cryptographic voting schemes issue a receipt to the voter that encodes the voter's choice. The difficulty is that the voter must be convinced that their choice was encoded correctly, and at the same time the receipt must not give any information about the voter's choice to a third party. Cryptographic voting schemes use a variety of methods to ensure both at the same time.

The Farnel schemes use a different approach. Instead of handing the voter a receipt of their own vote for verification, each voter verifies the vote of another voter or other voters. This solves the problem of receipt-freeness without using cryptography.

The name originates from the Portuguese word for basket and denotes the concept of a special ballot box introduced by Custódio [Cus01]. The exact properties of a Farnel ballot box differ for different schemes using such a Farnel ballot box. The most important property that is common to all version is that the Farnel ballot box is able to receive a ballot, shuffle its content and hand out one or more ballots or copies of ballots that were cast previously. It is also prepared with votes before the voting phase which are then subtracted during the tally phase.

In the original description of the Farnel voting scheme [ADC02], the Farnel ballot box receives the plaintext ballot of a voter, shuffles and returns a random ballot that was cast previously. The voter then signs the ballot and casts it into a second ballot box. After the voting phase the remaining ballots in the Farnel ballot box are signed by the voting authority and also included in the second ballot box. For the tally, the second ballot box is opened, all ballots are published and counted. To form the final tally the votes of the ballots used to initialize the Farnel ballot box are subtracted. Each voter is able to verify that all ballots are signed either by an eligible voter or by the voting authority. This scheme does not offer verifiability of the result, this still depends on an honestly behaving voting authority.

Rivest and Smith proposed a similar technique in [RS07] to obtain Twin, a simple protocol with its security and receipt-freeness essentially based on a Farnel-like ballot box. The ballot is a simple paper ballot without any additional information except a unique ID hidden under a scratch-off coating. The voter marks the ballot and casts it into the ballot box. The ballot box removes the scratch-off coating and issues a previously cast ballot as a so-called floating receipt to the voter.

Araújo, Custódio and van de Graaf have proposed a voter-verifiable scheme based on the Farnel idea in [ACvdG07, ACvdG10]. The ballots of the improved scheme consist of two parts. One part contains the marking area with the candidates and a unique ID covered by a scratch-off coating, the other part only contains the same ID also covered by a scratch-off coating. Both parts are separated by a perforation that allows easy detachment. During the voting process the voter receives a ballot, removes the scratch-off coating to verify that both IDs are identical and marks the ballot. Then the voter separates the two parts of the ballot, casts the part containing the marking area into one ballot box and the second one into a Farnel box. Then the voter receives copies of several other slips containing IDs from the Farnel box. The voter is now able to verify that the ballots corresponding to those slips are published by checking the IDs.

Araújo and Ryan published another cryptographic voting scheme in [AR08b, AR08a] that combines a Prêt à Voter style receipt with a Farnel ballot box.

2.5.2 Votegrity

Chaum proposed the use of visual cryptography in [Cha02, Cha04] for the generation of receipts. The resulting cryptographic voting scheme, Votegrity, was the first scheme that offered human-verifiable correctness. The idea of Votegrity is to print a receipt onto two layers so that both layers together contain the information that allows the voter to verify that it contains the correct vote. When the voter leaves the voting booth the two layers are separated and one is shredded while the other is cast and a copy is given to the voter as receipt.

Chaum suggests using visual cryptography described by Naor and Shamir in [NS95]

to construct a two-layer receipt. This method is similar to the secret sharing scheme using a one-time pad. Each layer of the receipt contains a share of the information of how the voter voted. The information is printed onto the two transparent layers so that when overlaying both the information becomes readable to the voter. When one of the two layers is destroyed, the information of how the voter voted is no longer visible. But the remaining layer, a copy of which the voter receives as a receipt, also contains information the voting authority uses to reconstruct the vote during the tally phase. To prevent any single party from directly breaking voter privacy, the decryption process is spread among several trustees. Each trustee takes the output of the last trustee as input (the first one taking the receipts as input) and performs one step of the decryption process. Then the trustee shuffles all receipt before handing them to the next trustee. If at least one trustee is honest, the connection between the receipt and the unencrypted vote is hidden. When the last trustee decrypted all receipts, the content of each vote is readable and the tally becomes obvious.

For the verification the voter checks the correctly formed receipt during the voting phase. This is easy due to the use of visual cryptography. After the voting phase each voter is able to check that their receipt was published correctly. For the tally each trustee provides a proof that the encryption and shuffling step was conducted correctly. For this each trustee shows for a portion of the input receipts how they were mixed and that for those the partial decryption was conducted correctly.

2.5.3 MarkPledge

Neff described the cryptographic voting scheme MarkPledge in [Nef04] that uses a voting machine to record the voter's choice and generate a receipt. The receipt contains a ciphertext for each candidate so that the ciphertext for the elected candidate is an encryption of 1, the other ciphertexts all are encryptions of 0s. The voting machine includes a zero-knowledge proof that the ciphertext of the elected candidate is an encryption of 1 as well as equivalent simulated proofs for the other ciphertexts. The difference between the proof for the elected candidate and the other candidates is that the voter saw a pledge for the ciphertext representing the vote for the elected candidate before entering the challenge.

MarkPledge uses a special form of encryption to encrypt single bits. An encryption for 0 consists of k pairs of encryptions $[\text{Enc}(u_1)], [\text{Enc}(v_1)], \ldots, [\text{Enc}(u_k)], [\text{Enc}(v_k)]$ with $u_i, v_i \in \{0, 1\}$ and $u_i \oplus v_i = 1$. This means that each pair consists of one encrypted 0 and one encrypted 1. The encryption for 1 consists of k pairs of ciphertexts as well, but for those $u_i \oplus v_i = 0$ for all i. This means that either both ciphertexts of a pair contain 0 or both contain 1. MarkPledge uses the fact that there exists a simple 3-round zero-knowledge proof that an encryption contains a 1.

For the proof that the encryption $c = [\text{Enc}(u_1)], [\text{Enc}(v_1)], \ldots, [\text{Enc}(u_k)], [\text{Enc}(v_k)]$ contains 1, prover $\mathcal{P}$ first sends the string $s = u_1, \ldots, u_k$ to verifier $\mathcal{V}$. $\mathcal{V}$ sends a k-bit string $b = b_1, \ldots, b_k$ as challenge. $\mathcal{P}$ now sends the randomness used for the ith sub-ciphertext, if b_i is 0 the randomness used to encrypt u_i and if b_i is 1 the randomness used to encrypt v_i. This way $\mathcal{V}$ is able to check whether the sub-ciphertext contains u_i. If the encryption contains 1, $u_i = v_i$ for all i, so the content of the sub-ciphertext checked by $\mathcal{V}$ is always equal to the corresponding

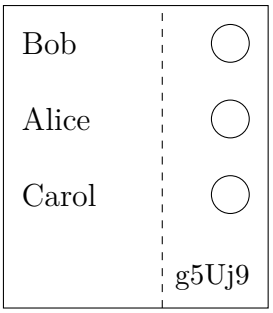


Figure 2.2: A ballot for an election employing Prêt à Voter. The ballot consists of two parts, separated by a perforation. The three candidates are printed in permuted order on the left part. The voter marks the ballot on the right part.

bit of s. If the encryption contains 0, $u_i = \bar{v}_i$ for all i, so the verification fails approximately half of the time.

This is zero-knowledge proof so a simulator $\mathcal{S}$ exists that is able to simulate an authentic looking transcript of a protocol run in which $\mathcal{V}$ accepts. The construction of $\mathcal{S}$ is straightforward: $\mathcal{S}$ first chooses the challenge string b and then constructs s accordingly so that $s_i = u_i$ if $b_i = u_i$ and $s_i = v_i$ if $b_i = 1$.

Adida and Neff describe an enhancement for the original version in [AN06]. In 2009, Adida and Neff published MarkPledge2 in [AN09] as a version of MarkPledge that is resistant to covert channels in the randomness used for the encryptions.

2.5.4 Prêt à Voter

Chaum, Ryan and Schneider described Prêt à Voter, a cryptographic voting scheme using paper ballots in [CRS04, CRS05]. The name translates to "ready to vote". Prêt à Voter uses special paper ballots that consist of two parts separated by a perforation that allows easy separation as shown in Figure 2.2. The left part of the ballot contains the candidate list in permuted order. This permutation is different for each ballot. The right part contains the marking areas for each candidate as well as a string called *onion* that contains the information about the permutation in encrypted form.

During the voting process the voter marks the ballot on the marking area on the right part of the ballot. Before casting the vote, the left part of the ballot containing the permuted candidate list is detached and destroyed. Without this permutation the voter's choice is no longer discernible. The remaining part of the ballot is copied, the copy is handed to the voter as receipt, and then cast into the ballot box. After the voting phase all ballots are shuffled and decrypted. The shuffling and decryption is conducted by several trustees (called *tellers*) that one after another take the encrypted ballots, remove one layer of encryption, shuffle the resulting ballots and hand it to the next teller.

For the verification the voter first checks that the onion correctly contains the permutation on the left part of the ballot. This is done using a simple cut-and-choose protocol: The voter receives two ballots and may challenge one. For the challenged ballot the correctness of the onion is then shown and the voter may use the other ballot for the voting process. After the voting phase, the voter is able to check that their receipt was published correctly. The correct decryption

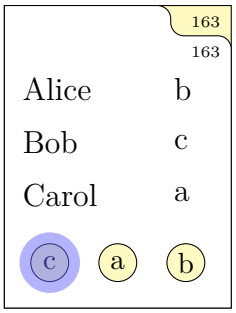


Figure 2.3: A Punchscan ballot for an election with three candidates. The ballot consists of two parts, the top part (white) has holes through which the bottom part (yellow) is visible. The ballot is marked for candidate 'Bob'.

of the ballots by the tellers is proven by randomized partial checking.

2.5.5 Punchscan

Chaum developed Punchscan, a cryptographic voting scheme using two-layer paper ballots, and described by Popoveniuc and Hosp in [PH06a, PH06b, PH10].

Similiar to Prêt à Voter, Punchscan uses a paper ballot and a permutation of the candidates to encrypt the voter's choice. The ballot consists of to layers, the top layer has several holes through which the bottom layer is visible (see Figure 2.3). Both layers also contain an identical ID. The top layer contains the candidates in arbitrary order (i. e. the order may be identical for all ballots in contrast to Prêt à Voter) and a letter next to each candidate. There is one hole in the top layer for each candidate and in each area of the bottom layer visible through such a hole there is one letter printed. To mark a ballot for a certain candidate, the voter looks at the letter standing next to the candidate and finds the corresponding area on the bottom layer visible through a hole in the top layer. The voter now marks both the area of the bottom layer with the correct letter as well as the area of the top layer around the hole through which the letter is visible. This may be done using a Bingo dauber. The voter now separates both layers and chooses one as receipt. This layer is copied, the copy handed to the voter as receipt and the original cast into the ballot box. The other layer is destroyed.

The ID on the remaining layer contains sufficient information to construct the voter's choice. Similar to Prêt à Voter this is done by tellers that decrypt and shuffle the ballots to form the tally. For verification, each voter may check that their receipt was published correctly and that the proofs for the shuffles and decryption steps are correct.

2.5.6 Voting Scheme by Moran and Naor

Moran and Naor presented the first universally verifiable voting scheme [MN06] with its security based on the assumption that a non-interactive commitment scheme exists. It uses a voting machine to receive the voter's choice and generate a receipt. The underlying idea is that the voting machine prints a commitment to the candidate of the voter's choice and then proves that the content of the commitment is the name of the candidate the voter voted for. For this the scheme uses a zero knowledge proof (cf. Section 2.4.3).

One big advantage of the voting scheme by Moran and Naor is that it requires almost no specific preparation. The voting authority has to input the candidate list into the voting machine and the public information required for the commitment scheme used. Moran and Naor describe their voting scheme using UHDLCs, so the voting machine requires a description of the group $(\mathbb{G}, \cdot)$ and two generators g and h of $\mathbb{G}$.

When a voter chooses a candidate at the voting machine, the machine prints a commitment to the candidate on the receipt. Then the voter is asked to enter random strings for all candidates but the one the voter voted for. The voting machine generates challenges from these strings using a hash function and generates new commitments for all candidates. These new commitments are then printed onto the receipt without showing them to the voter (the part of the receipt containing the commitments is covered with an opaque shield). The voter must be able to verify that the commitments were printed (for correctness) but not able to see them (for receipt-freeness).

After the commitments were printed, the voter enters a random string for the candidate the voter voted for. The voting machine then prints proofs that the new commitments are either commitments for the corresponding candidate or that they are a rerandomization of the first commitment (containing the voter's choice). The challenges entered by the voter determine what is proven. Since the challenges for the candidates the voter did not vote for were entered before the commitments were printed, the voting machine prepared these new commitments accordingly. But as the challenges for the candidate the voter voted for was entered after the new commitments were printed, this gives a zero-knowledge proof that the first commitment indeed contains the voter's choice.

For the tally the voting authority takes the first commitment of each receipt and uses a shuffle of known content to prove that the tally corresponds to the content of these commitments without giving any additional information about the choice of a single voter.

Bohli, Müller-Quade and Röhrich described an attack on the voting scheme by Moran and Naor that precludes coercion resistance in [BMQR07a]. For this so-called babble attack, the coerced voter receives instructions from the adversary inside the voting booth while interacting with the voting machine.

2.5.7 ThreeBallot

ThreeBallot was first described by Rivest in [Riv06] as an end-to-end-verifiable voting scheme without cryptography. The ThreeBallot scheme uses special paper ballots, each of which is divided into three identical parts (see Figure 2.4). Each part contains a complete list of all candidates and a marking area. The three parts are connected with a perforation that allows the voter to easily separate the three parts. All three parts carry a unique ID, the IDs are chosen independently for each part so it is not recognizable which parts belong together once the three parts are separated.

To cast a vote, the voter marks each candidate on one of the three parts and the candidate of their choice on a second part. After the voting process, the ballot must be presented to a checker machine. The checker machine ensures that a ballot is marked correctly. The three parts are separated, one is copied and given to the voter as receipt. As every part may contain a mark for every candidate,

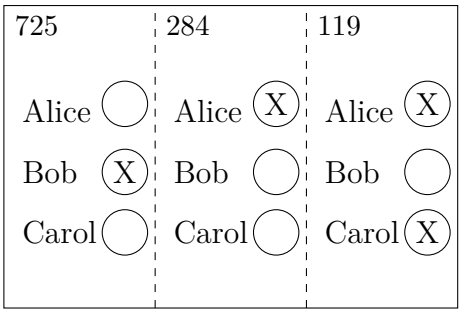


Figure 2.4: A ballot for an election employing ThreeBallot. The ballot consists of three parts that are separated by a perforation. Each part contains a complete list of candidates and a marking area as well as unique ID. This ballot is marked for candidate 'Alice'.

whether or not the voter voted for him, the voter is unable to use the receipt to prove how they voted. Only the the checker machine is able to detect undervoting or overvoting during the voting process, once the three parts are separated this is impossible (as their association must be kept secret for ballot secrecy).

After the voting phase, all ballot parts are published. Each voter checks whether their receipt is published. Under the assumption that the adversary does not know which part of a ballot the voter chose as receipt, the manipulation of a vote is detectable with probability $\frac{1}{3}$, as the manipulation changes at least one ballot part which is chosen as receipt with probability $\frac{1}{3}$.

Küsters, Truderung and Vogt have identified a severe weakness of ThreeBallot in [KTV11]. If the checker machine that enforces that only well-formed ballots are cast is corrupted, a corrupted voter is able to cast a negative vote for a candidate by simply not marking this candidate any of the three parts.

2.5.8 Scantegrity and Scantegrity II

Scantegrity is a cryptographic voting scheme proposed by Chaum et al. in [CEC+08]. It uses paper ballots that include a unique ID and a short code (typically three characters) for each candidate. The voting process for an election using Scantegrity is almost identical to an election with traditional paper ballots. The only difference is that the voter, after marking their choice, records the code associated with the selected candidate. This code forms the receipt together with the ID of the ballot. After the voting phase the voting authority publishes for each ballot the code of the elected candidate. To verify the correct inclusion of their vote into the tally the voter simply checks that the correct code was published.

Scantegrity Invisible Ink (or Scantegrity II) is an improvement of the Scantegrity scheme. It uses invisible ink to conceal the codes for each candidate. During the voting process the voter marks the candidate of their choice by filling out a bubble next to the candidate with a special pen. This pen contains activation ink that reacts with the invisible ink printed in the bubble. There are two different inks, a reactive ink that turns black when coming in contact with activation ink, and a dummy ink that stays the same. These two inks are used to print the codes for each candidate inside the bubbles the voter fills to vote for the candidate. This way the voter only learns the code for the candidate the voter voted for. As with Scantegrity, the voter records this code on a paper slip that also contains the ID

of the ballot. This acts as a receipt and enables the voter to check that the correct code is published for their ballot. If this is not the case, the knowledge of a valid code acts as proof that the vote of the voter was manipulated.

In [EHH11], Essex, Henrich and Hengartner proposed to create the ballots in a shared way, so that no single party knows the connection between a code and a candidate.

3. Bingo Voting

Bingo Voting is a cryptographic voting scheme that was developed by Bohli, Müller-Quade and Röhrich and first described in [BMQR07a]. It employs a voting machine (with arbitrary user interface, see Section 6.4.1) and a *trusted random number generator*. The original scheme claims end-to-end security (cf. Section 2.3) with the correctness relying only on the security of the trusted random number generator (see Section 6.3 for details about the relations between security properties and the required assumptions). The voter receives a receipt after casting their vote. The scheme uses random numbers to encode the vote on the receipt so that no third party learns what vote the receipt stands for (making Bingo Voting *receipt-free*). The voter is able to verify that the receipt correctly encodes their vote during the voting process (*cast-as-intended*). After the voting phase copies of all receipts issued are published, allowing the voter to verify that their vote was included in the tally correctly (*counted-as-cast*).

This chapter describes the original version of Bingo Voting as described in [BMQR07a, BMQR07b]. For this explanation we will consider a simple election with one voting machine and n candidates of which the voter elects one (one vote per voter), i. e. a 1-out-of-n election. Section 5.1 discusses the changes necessary to use Bingo Voting for a more complex election. Section 3.1 presents the central idea of Bingo Voting as well as several important concepts, notions and primitives used by the original scheme. Many of these concepts and primitives are used by the advanced scheme (described in Chapter 5) as well. Section 3.2 gives a short overview of Bingo Voting and illustrates the basic idea as well as the voting and verification procedure on a conceptual level. Section 3.3 describes the voting process in detail and gives an in-depth explanation of the processes happening during the different phases of the election. Section 3.4 explains the verification process for each voter as well as the public (i. e. individual and global verification). While this chapter describes the original scheme, we employ the same terminology for the improved version presented in Chapter 5.

3.1 Important Concepts

The central idea of Bingo Voting is that the voting machine encodes the voter's choice in their receipt using random numbers. Each receipt in this election contains

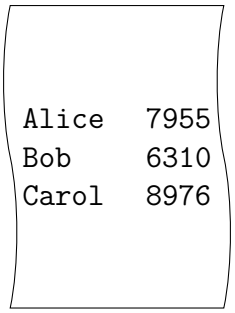


Figure 3.1: A Bingo Voting receipt in a simple election with the three candidates (`Alice`, `Bob`, and `Carol`) and one vote per voter. The random numbers printed next to each candidate encode the voter's choice.

each candidate and one random number assigned to it (e. g. simply written next to the candidate). Figure 3.1 shows an example of a receipt for an election with three candidates and one vote per voter.

There are two types of random numbers, *dummy random numbers* generated *before* the voting phase and *fresh random numbers* that are generated *during* the voting process by the trusted random number generator. The random number used to denote the voter's choice is the fresh random number generated and displayed by the trusted random number generator inside the voting booth. All other random numbers on the receipt are dummy random numbers.

To turn this idea into a verifiable voting scheme we require several additional mechanics. The remainder of this section will describe the necessary primitives and concepts and introduces several notions specific to Bingo Voting.

3.1.1 Trusted Random Number Generator

Bingo Voting requires a trusted random number generator to convincingly generate fresh random numbers during the voting process in front of the eyes of the voter, display it in a human readable form and send it to the voting machine.

The trusted random number generator is the anchor of confidence. Section 6.3 discusses the assumptions required for the different security properties in detail. The quintessence is that the correctness of Bingo Voting is based on the unpredictability of the trusted random number generator and the binding property of the commitments (see Section 3.1.2) used. The receipt-freeness is based on the indistinguishability between dummy random numbers and fresh random numbers as well as on the isolation assumption of the voting booth (including the voting machine the voters use to cast their votes).

The original work on Bingo Voting [BMQR07a] proposes using randomness from a mechanical source, for example a bingo cage, to generate the random numbers necessary for the voting scheme. Section 4.3 describes the trusted random number generator employed in the real-world election using Bingo Voting. This trusted random number generator was not mechanical but used a random number generator on a signature card. Section 6.2.4 discusses different forms of random number generators and their impact on the security and usability of Bingo Voting.

3.1.2 Commitments for Bingo Voting

The security of Bingo Voting relies on the distinction between *fresh* random numbers, generated during the voting process (see Section 3.1.1), and *dummy* random numbers (see Section 3.1.3). For correctness, the dummy random numbers must be verifiably fixed before the voting phase starts. For receipt-freeness, the fact which random number on a receipt was fresh and which was generated before the voting phase must be kept secret.

To ensure both properties, Bingo Voting employs a commitment scheme (see Section 2.4.2 for an introduction to commitments) called *unconditionally hiding discrete logarithm commitment* (UHDLC) that was first described in [CDvdG87] but is often referred to as Pedersen Commitments [Ped92]. UHDLCs are unconditionally hiding. The binding property is based on the discrete logarithm assumption.

During the preparation phase, the voting authority publishes dummy votes consisting of a pair of commitments, one to a dummy random number and one to the corresponding candidate (see Section 3.1.3 for details). The *hiding property* of the commitment scheme prevents the adversary $\mathcal{A}$ from learning the dummy random number as well as from gaining any information about which candidate the dummy vote is for. The *binding property* guarantees that the voting authority may not change the dummy random number or the candidate of the dummy vote. See Section 6.3 for a detailed discussion of the security properties of Bingo Voting and which assumptions they are based on.

In addition to these two security properties Bingo Voting makes heavy use of a third property of UHDLCs. At several points the voting authority has to prove that a set of plaintexts is indeed the content of a set of commitments without divulging any further information about which commitment contains which plaintext. This is called a proof of a shuffle with known content. The use of UHDLCs allows the employment of randomized partial checking (RPC). RPC proofs are zero-knowledge proofs (cf. Section 2.4.3) that utilize the fact that UHDLCs are rerandomizable.

Section 2.4.2.1 gives a detailed explanation of UHDLCs. Section 2.4.2.2 explains how a UHDLC is rerandomized. Section 2.4.4.1 describes the zero-knowledge proof used in the original Bingo Voting scheme that utilizes this property to verifiably prove the correctness of an election without compromising voter privacy.

3.1.3 Dummy Votes

Bingo Voting uses *dummy votes* as the central concept to keep track of the votes cast. Each dummy vote consists of a pair of commitments (cf. Section 2.4.2), one to a random number `r` and one to a candidate `Cand`. We call this pair

$$\left(\textsc{Commit}\left(\mathtt{r}\right), \textsc{Commit}\left(\mathtt{Cand}\right)\right) = \left(g^{\mathtt{r}} \cdot h^{r_{\mathtt{r}}}, g^{\mathtt{Cand}} \cdot h^{r_{\mathtt{Cand}}}\right) \tag{3.1}$$

a *dummy vote* for candidate `Cand` with dummy random number `r`.

Each dummy vote represents a potential vote. During the preparation phase, the voting authority generates v dummy votes for each candidate where v is the number of voters in the election. For the correctness of the election it is essential that the number of dummy votes generated is the same for each candidate. The voting authority proves this using a zero-knowledge proof that does not reveal the dummy random numbers at all nor which dummy vote belongs to which candidate.

This is important to ensure the receipt-freeness of Bingo Voting. Section 6.3 discusses this in detail.

It is also important for correctness that all random numbers are unique (or that there are only very few collisions). Since each collision means a potentially changed vote, the number of collisions that is acceptable is determined by the number of votes that may be changed without changing in the result of the election. As all dummy random numbers become visible during the tally phase this is easy to verify (see Section 3.4.2). Section 6.3.5 discusses the consequences of collisions in detail. Section 5.2 proposes a change that makes collisions less likely without increasing the length of random numbers (which affects usability, cf. Section 6.4.1).

The role of dummy votes in Bingo Voting is described briefly in Section 3.2 and in detail in Section 3.3.

3.2 Bingo Voting in a Nutshell

This section gives a short overview over the Bingo Voting protocol and points out the central mechanisms that ensure Security considerations and details are only sketched or omitted completely in this section to concentrate on the central idea. In this section we will look at a simple election with v voters, n candidates and one vote per voter (1-out-of-n election).

As mentioned in Section 3.1.3, Bingo Voting is based on the idea of dummy votes representing potential votes. In the preparation phase, the same number of dummy votes is allotted to each candidate. During the voting phase, whenever a voter votes for candidate `Cand`, each candidate except `Cand` loses a dummy vote. This means that candidate `Cand` now has one dummy vote more compared to all other candidates. For this reason the number of dummy votes remaining for each candidate directly results in the tally if at the beginning each candidate had as many dummy votes as there were voters that cast their votes during the voting phase.

For the correctness, we have to ensure two things. The first thing the voting authority has to prove is that each candidate has initially received the same number of dummy votes. This number is typically identical to the number of voters expected to vote during the voting phase. The fact that normally this number is bigger than the number of voters that actually cast a vote is not an issue. For the final tally one simply subtracts the difference between the number of votes cast and the number of dummy votes per candidate issued during the preparation phase. The second thing to prove is that whenever a voter casts a vote for a candidate, each other candidate loses exactly one dummy vote.

To prove that each candidate has received the same number of dummy votes the voting authority publishes the dummy votes of all candidates together with a proof. The proof shows that the content of the second part of each dummy vote (containing the commitment to the candidate) is equal to a list of candidate names on which each candidate name appears equally often. The proof that for each voter casting a vote every candidate except the one elected loses a vote takes several steps. To generate the receipt the voting machine takes the dummy random number of each dummy vote lost and prints it next to the corresponding candidate. The random number printed next to the candidate the voter cast their vote for is the random number generated by the trusted random number generator. The voter is able to check that the receipt correctly encodes their choice by comparing

the display of the trusted random number generator with the random number printed next to their candidate. The second step in the proof is that the voting authority proves that each but one random number on the receipt is a dummy random number. The voter knows that the random number for the candidate they voted for is highly unlikely to be a dummy random number as it was generated by the trusted random number generator during the voting phase. The conclusion is that all other candidates must have lost one dummy vote.

Section 3.3 describes the voting process with Bingo Voting in detail. Section 3.4 presents the verification process and explains how each voter is able to check the two things mentioned above and therefore the correctness of the election.

3.3 Bingo Voting Protocol

This section describes in detail how the Bingo Voting election scheme as presented in [BMQR07a] is used in an election. In this description we concentrate on details specific to Bingo Voting and will mention generic steps in an election only when relevant to Bingo Voting. Section 3.3.1 presents the preparations necessary for the election and everything that has to be done before the actual voting phase. Section 3.3.2 describes the voting phase, how a voter actually casts their vote and the first part of the individual verification process. Section 3.3.3 explains how the tally is generated and what is published after the voting phase for the second part of the individual and the global verification.

For the explanation in this section we assume that each voter only has one vote and that the election uses only a single voting machine. See Section 5.1 on how to expand Bingo Voting for larger and more complex elections.

Bingo Voting employs a voting machine to manage the dummy votes, receive the vote cast by the voter and generate the receipt. The original paper suggests that the voting machine also generates the dummy votes. While this is not essential (as we will see in Section 4.2.4), we adopt this concept for the explanation in this section. How the voting machine interacts with the voter exactly is not fundamental for Bingo Voting as long as the voter is able to enter their choice unambiguously (cf. Section 6.4.1). For this explanation we will that assume the voting machine has an appropriate input device on which the voter marks the candidate of their choice. See Section 4.1.4 on how a prototype of a Bingo Voting voting machine was built.

3.3.1 Preparation Phase

Before the voting phase, the election authority publishes the required parameters of the election. This includes the list of candidates. For an election employing Bingo Voting, the voting authority also has to generate and publish the set of dummy votes. The original paper [BMQR07a] assumes that the voting machine itself generates the dummy votes using the same trusted random number generator that is also used during the election. The voting authority enters the candidate list and the number of voters into the voting machine, the voting machine generates one dummy random number for each candidate and each voter, saves the sets of dummy random numbers for each candidate together with the randomness used for the commitments (this is required to generate receipts and proofs of correctness before and after the voting phase) and finally generates and outputs the dummy votes.

In addition to the set of dummy votes the voting machine generates a proof using randomized partial checking to show that each candidate has received the same number of dummy votes. This proof takes the second commitment of each dummy vote (containing the candidate) and proves that the content of this set of commitments is equal to a list of candidates on which each candidate appears the same number of times. The details of this proof are presented in Section 3.4.1.

Once the voting authority has informed all eligible voters about the election and published all necessary data, the preparation phase ends. Only after the preparation phase has ended the voting phase may start.

3.3.2 Voting Phase

During the voting process, each voter casts their vote and receives a receipt. The receipt is generated by the voting machine directly after the vote was cast. With this receipt a voter is able to verify that their vote was counted correctly. This section describes the voting process in detail.

Before the actual voting process, the voting authority has to ensure that the voter is indeed eligible to vote. This procedure is independent of the voting scheme used and out of scope of this work. The authenticated voter enters the voting booth. The voting booth guarantees privacy during the voting process, independently of the voting scheme used. See Section 6.2 for a detailed discussion on the necessary properties of a voting booth.

As mentioned above, Bingo Voting only requires that the vote must be input into the voting machine during the voting process. In addition to the voting machine (including the input device), two more things are necessary for Bingo Voting: a printer and the trusted random number generator with a display, connected to the voting machine and inactive before the voter has cast their vote. Only after the voting machine has received the vote, the voter is actually confronted with anything that is specific to Bingo Voting (as argued in Section 6.4.1).

After the vote was cast the voting machine generates a receipt. For this the trusted random number generator generates one fresh random number. The computer forms the receipt by printing every candidate together with a random number. The random numbers printed next to candidate $\texttt{Cand}_i$ is either the random number drawn by the trusted random number generator (if the voter voted for $\texttt{Cand}_i$) or a dummy random number from an unused dummy vote for $\texttt{Cand}_i$ (if the voter did not vote for $\texttt{Cand}_i$). The dummy votes used for the other candidates are marked on the list as 'used' and are not used for any other receipt.

The receipt is printed and handed to the voter. The voter may then check whether the receipt encodes their choice correctly by comparing the random number next to the candidate of their choice with the display of the trusted random number generator. See Figure 3.2 for an example of the view of a voter checking the receipt in the voting booth. If satisfied, the voter leaves the voting booth and ends the voting process. The original paper does not include means for dispute resolution, see Section 5.4 for a discussion about this.

The voting phase ends when each eligible voter has cast a vote or chosen to abstain. After the voting phase ends the tally phase begins.

3.3.3 Tally Phase

After the voting phase the voting authority determines and publishes the tally and the result of the election together with the proofs required for the verification

of the tally. Please note that for this explanation we will only consider the tally as we assume that the election result is determined by a deterministic process from the tally.

For an election employing Bingo Voting the voting authority publishes three sets of data during the tally phase:

- The voting authority publishes a digital copy of each receipt issued during the voting phase. Each voter is able to check that their receipt was published correctly.
- The voting authority proves for each receipt published that it contains the correct number of dummy random numbers, i. e. that all but one random number on the receipt is a dummy random number. Each such proof is a non-interactive zero-knowledge proof using randomized partial checking that shows additionally that each dummy random number on the receipt is associated with the correct candidate (the same candidate the dummy vote it originates from belongs to).
- The voting authority opens all dummy votes that were not used during the voting process to form a receipt. We call these unused dummy votes *remaining dummy votes* for the candidate indicated by the (now opened) second commitment of the dummy vote.

The number of the remaining dummy votes for each candidate directly corresponds to the number of votes this candidate received during the voting phase (and thus the tally).

To calculate the tally from the number of remaining dummy votes the voting authority simply has to subtract the number of non-voters, i. e. the difference between the number of voters the voting authority prepared dummy votes for and the number of voters that actually have cast a vote. These numbers are directly computable from the number of published dummy votes in the preparation phase (which is equal to the number of voters times the number of candidates) and the number of published receipts (which is equal to the number of votes cast).

3.4 Verification

One very prominent feature of cryptographic voting schemes is that each voter is able to verify that the tally is correct. For Bingo Voting this verification consists of two parts: The first part, in which the voter checks that their vote was included in the tally, we call the *individual verification* and discuss in Section 3.4.1. The second part, for which any auditor checks that the tally is indeed the sum of all votes cast, we call *global verification* and discuss in Section 3.4.2.

For the verification of an election using Bingo Voting the voting authority has to prove two things:

- Each candidate had the same number of dummy votes before the voting phase.
- For each voter who voted, each candidate who did not receive their vote lost one dummy vote.

These two things are part of the individual as well as the global verification process. This section describes both parts of the verification process in detail.

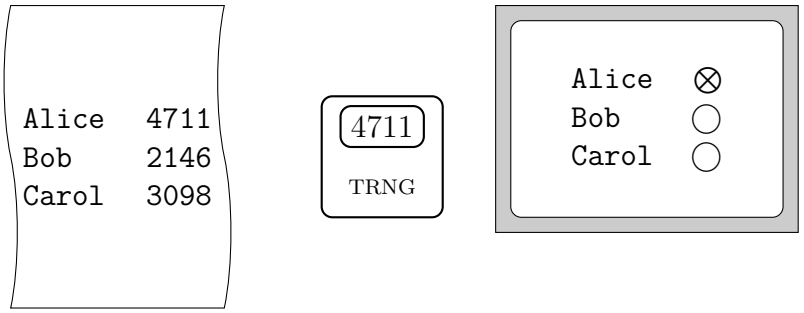


Figure 3.2: After casting their vote using the voting machine (right) the voter receives a receipt (left). The voter is now able to verify that the receipt correctly encodes their choice by comparing the random number displayed by the trusted random number generator (middle) with the random number corresponding to their choice on the receipt. In this example the receipt correctly encodes the vote for `Alice`.

3.4.1 Individual Verification

For the *individual verification*, each voter verifies that their vote was included in the tally correctly. The individual verification consists of two steps. During the voting process, the voter verifies that the receipt correctly encodes their choice. For this, the voter compares the receipt with the display of the trusted random number generator and checks that the random number next to the candidate the voter voted for is the same as on the display (see Figure 3.2). This is done inside the voting booth directly after the voter has cast their vote at the only moment of the whole election during which the receipt is readable (without the secret knowledge of the voting machine). This is also the only step the voter is required to do themselves. If the fresh random number is associated correctly with the candidate the voter intended to vote for, the voting machine has correctly received and recorded the vote. We say the vote was *cast-as-intended*. This step is a simple comparison of two random numbers and the only step which requires knowledge of the choice the receipt represents.

After the voting phase, the voter checks that their receipt is correctly published. If the receipt issued during the voting process is identical to the receipt published afterwards by the voting authority, the vote was included in the tally correctly. We say the vote is *counted-as-cast*.

The second step, ensuring that the vote was counted as cast, the voter may delegate by handing their receipt (or just a copy of the receipt) to one or several auditors. Those might be family members, representatives of political parties or external auditors. To ensure that the vote corresponding to a certain receipt was included in the tally correctly it is only necessary to check that the published receipt is identical to the hardcopy issued during the voting phase. It is not necessary to know which candidate the vote was cast for.

While it is not necessary to know what vote the receipt encodes in order to check the correct publication by the voting authority, it is necessary to possess the receipt (or at least an identical copy) since the assumption that a human voter is able to remember the complete content of receipt is unrealistic. The assumption that the correct publication of each receipt is checked is a very strong assumption. It is also

unnecessary if the adversary is unable to predict which receipts are checked and if a significant percentage is checked. If this is the case the adversary risks detection with each altered vote, making an undetected manipulation of a large number of votes unlikely. Section 5.6 describes the receipt stealing attack which makes the assumption that adversary does not know which receipt is checked invalid and presents a countermeasure.

Both steps of the individual verification process are comparisons. The first step inside the voting booth is a simple comparison of two random numbers. The second comparison is more involved as it requires the comparison of a whole receipt, but it may be delegated to an arbitrary number of auditors by issuing copies of the receipt.

3.4.2 Global Verification

The individual verification ensures that the vote of each voter was cast as intended and counted as cast. For this, however, we have to assume that each voter checked (directly or indirectly) the steps described in Section 3.4.1 or that the adversary is unable to predict which receipts are checked. Section 6.3 discusses these assumptions in detail.

Bingo Voting uses dummy votes to internally track the votes cast and to produce a verifiable tally. At the beginning each candidate has the same number of dummy votes and during each voter's voting process each candidate, except the one the voter voted for, loses a dummy vote. Therefore the number of remaining dummy votes directly translates to the tally (see Section 3.3.3). The individual verification convinces the voter that the candidate they voted for has not lost a dummy vote and that the vote was included in the tally.

For the global verification the voting authority has to show four things:

- Each candidate had the same number of dummy votes in the beginning.
- Each receipt is well-formed and contains not more than one dummy vote of each candidate. This is equivalent to proving that each candidate lost at most one dummy vote for each voter.
- Each dummy vote was either used, i. e. written on a receipt, or not used, i. e. opened and published as part of the tally.
- There are only statistically insignificant few collisions, i. e. identical (dummy) random numbers for different candidates.

To verify these four things it is not necessary to be an eligible voter or having participated in the election at all. The public data published by the voting authority is sufficient.

The first proof showing that each candidate has received the same number of dummy votes before the voting phase is a simple proof of a shuffle of known content for a set of commitments (see Section 2.4.4.1). The voting authority acts as prover $\mathcal{P}$, the public input is the list of all commitments to candidates that are part of dummy votes

$$C_{\texttt{Cand}} = \{c_{\texttt{Cand}} \,|\, d = (c_r, c_{\texttt{Cand}}) \text{ is a dummy vote}\} \tag{3.2}$$

and the list M containing each candidate v times with v being the number of voters. The proof uses randomized partial checking to show that the content of the commitments in $C_{\texttt{Cand}}$ is identical to M. Please note that it is actually possible to do this after the voting phase as shown in Section 5.5, reducing the size of the proof.

The second proof published by the voting authority shows that each receipt is well-formed. For this the voting authority takes for each receipt every dummy vote used on this receipt together with a new pair of commitments. This new pair contains the random number generated by the trusted random number generator during the voting process and the candidate the voter voted for. The voting authority then shows that the content of these commitments (the dummy votes and the new commitment) is identical to the content of the receipt. Using these pairs of commitments for a proof of a shuffle of known content the voting authority proves that all but one random number on the receipt are dummy random numbers and that each dummy random number is associated with the correct candidate.

The third thing is to show that each dummy vote was either used on a receipt or remained unused and contributed to the tally. This is actually quite easy to verify as each dummy vote published during the preparation phase must be either used for a proof that a receipt was well-formed or opened if it was unused on a receipt. It is also easy to check that each dummy vote was used on at most one receipt and not used on a receipt *and* marked as unused and opened.

The last thing missing is to verify that there are not more collisions than expected if all random numbers were drawn uniformly from the set of all possible random numbers. This is easy as all random numbers are public, either as part of a receipt or as opened dummy random numbers from unused dummy votes. We discuss the consequences of a collision in detail in Section 6.3.5. The quintessence is that a corrupted voting machine may be able to change a single vote when a collision occurs. The original description of Bingo Voting claims that collisions are negligible as they are effectively prevented by choosing longer random numbers. Unfortunately, longer random numbers make the comparison of the receipt with the trusted random number generator in the voting booth more demanding for the voter. Section 5.2 presents a change to the protocol that effectively reduces the occurrences of collisions that allow for an undetected manipulation of a vote.

Please note that the verification of the eligibility of each voter and that each voter only casts at most one vote is not part of the Bingo Voting voting scheme. While this an integral part for verifiable correctness, it is not part of most cryptographic voting schemes and is usually ensured by procedural measures. See Section 6.2.1 for a detailed discussion.

4. Real-World Experiences with Bingo Voting

There are many cryptographic voting schemes aiming at implementing a secure election (cf. Chapter 2.5). Unfortunately, only a few of those have demonstrated that they are employable in a real-world election [ECCP07, CCC+10]. Bingo Voting was employed in a small but complex election for a student parliament, demonstrating that it is indeed usable for real elections. This chapter presents the experiences from the deployment of Bingo Voting in the student parliament elections of the University of Karlsruhe in 2008. The results described here are published in [BHMQ+08].

Most elections, especially political elections, have to ensure accessibility to all voters. This important property, implied by the equality of all voters (so no voter may be discriminated against in terms of access to the election), is often considered afterwards when designing a cryptographic voting scheme. The main reason for this probably is that the exact criteria for accessibility of an election are hard to grasp and change over time (as new technology becomes widely-used and common). Another reason might be that most research in cryptography and cryptographic voting schemes model each party in a cryptographic protocol (in this case each voter participating in the election) as a Turing machine. Currently, good models for human behaviour in cryptographic protocols are still lacking. There are attempts to include specifics of human behaviour in the design of cryptographic voting schemes [Cha02, BLS+03, BHMQ+08, WH09], but approving a cryptographic voting scheme for use in public elections is an interdisciplinary effort that requires research after the security of the scheme has been shown.

The central goal of a cryptographic voting scheme must be to form a consensus among voters that the election result represents the will of all voters. A cryptographic voting scheme that is perfectly secure in theory may still be useless due to practical restrictions, usability issues or simply because it is not understood or accepted by voters. This makes demonstrations in real elections necessary for any cryptographic voting scheme to realistically assess the quality of a scheme. Many issues only arise when the theoretical scheme meets reality and real voters. Therefore, testing cryptographic voting schemes in real-world scenarios is fundamental

to prove their capability.

The student parliament elections of the University of Karlsruhe (now Karlsruhe Institute of Technology) in 2008 employed Bingo Voting as one possibility for voters to cast their votes. We implemented a prototype that was specifically constructed to meet the requirements of this election (which were changed slightly to allow for electronic voting machines as described in Section 4.1.3). The election and its requirements were complex and diverse. This made implementation difficult but at the same time showed that Bingo Voting is indeed practical for more intricate elections.

This chapter gives a detailed recount of the deployment of a Bingo Voting prototype at the student parliament election 2008 at the University of Karlsruhe. Section 4.1 describes the details of the election, especially the election mode that allowed voters to distribute up to five of their nine votes on a single candidate. The changes that were made to the original Bingo Voting protocol are explained in detail in Section 5.1. Section 4.2 presents the observations and experiences made during the election. Section 4.3 briefly describes the latest prototype that evolved from the voting machine used for this election. The work presented in this chapter was published in [BHMQ+08].

4.1 Election Details

The election of the student parliament of the Karlsruhe University (now Karlsruhe Institute of Technology) is an annual election for the representatives forming the student representative body. At the same time of the election of the student parliament there are elections for the women's representative and the representative of foreign students as well as for the student council for each faculty.

The elections took place concurrently on five consecutive days (Monday, January 14 2008 to Friday, January 18 2008). The voting machine had to manage fifteen different elections. A single voter was eligible for at least two and up to four elections. Each student was eligible to vote in the election for the student parliament (which was internally modelled as two distinct elections) and one out of eleven faculty student councils. Female and foreign students were eligible to additionally elect representatives of women and foreign students respectively.

Only a small part of the voters voted using the Bingo Voting prototype. The majority of the voters used traditional paper ballots. Potential reasons for this are discussed in Section 4.2.4.

The candidates for the student parliament were organized in lists and each voter has one vote for electing a list as well as several votes to distribute among the candidates to determine the candidate order within a list. The nature of this election made the voting process slightly complicated and consequently posed several challenges for the implementation of Bingo Voting used in the election. The other elections for the women's representative, the representative for foreign students and for the student council of each faculty were simple in comparison. For this reason we look at the details of the election for the student parliament in Section 4.1.1 and only briefly present the details of the other three elections in Section 4.1.2. Section 4.1.3 gives the details of the formal requirements for these elections and the changes made to the election regulations by the student parliament before the election to allow for electronic voting machines. Section 4.1.4 describes the implementation of the voting machine used for the election.

4.1.1 Student Parliament Election

The election for the student parliament was actually split into two distinct races that had to be handled as two different elections by the Bingo Voting voting machine. The voter first selected one of the nine lists and then had nine additional votes that they were allowed to distribute among the 72 candidates on the lists with a maximum of five votes per candidate. The first race was realized as a simple election with eleven candidates (the nine different lists plus 'abstention' and 'invalid', cf. Section 5.1.1) and one vote per voter. The second was realized as a distinct election with 74 candidates (72 candidates plus 'abstention' and 'invalid') and nine votes per voter, up to five of which a voter was allowed to cast for a single candidate.

These two races belonged together, there was in fact only a single paper ballot in the original election for both choices. For this reason the voting machine had to ensure that a voter did not vote in only one of the two races. See Section 4.1.4 for details.

4.1.2 Other Elections

The two elections for the representatives for women and foreign students were both simple elections with two (for the women's representative) and three (for the foreign student's representative) candidates respectively and one vote per voter.

The elections for the faculty student councils were different for each faculty. Each student was eligible for the election of their faculties student council and had to choose from five to ten candidates. Each voter had as many votes as there were candidates and allowed to freely distribute those votes among all candidates.

4.1.3 Formal Requirements

The members of the student parliament acted as voting authority. The student parliament had to change the election regulations, as the original version only considered elections with pen and paper and the wording was quite specific to only allow paper ballots. One example was that a voter, after the eligibility was confirmed, received a paper ballot to fill out, or that the paper ballot had to be folded before casting it into the ballot box. The changes required to allow casting a vote at a voting machine were minor and mostly generalized the wording of the election regulation to include electronic voting machines.

One important requirement was that the Bingo Voting scheme had to be able to allow everything that was possible with a paper ballot. This included partial and full abstention, casting invalid ballots and voting in different elections on different days.

The Bingo Voting election was treated as an additional ballot box. The regulations stipulated the publication of the result of each ballot box separately which made it possible to publish the information required for the verification process. Making the tally of the Bingo Voting election public is an integral part of the verification process (see Section 3.4), so if this was not allowed by the election regulations employing Bingo Voting would not have been possible.

4.1.4 Implementation of the Voting Machine

Michael Bär and Carmen Kempka (née Stüber) implemented the prototype, writing approximately 8000 lines of code in Java 1.6. For Michael Bär this was part

of his diploma thesis [Bär08]. The platform was a standard personal computer with Linux (SuSE 3.2) as operating system, 2 GB of RAM and an AMD dual core Opteron with 2.2 GHz CPU clock rate. We equipped the computer with an external chip card reader by ReinerSCT, a computer mouse as input device and a laser printer. The trustworthy random number generator was realised by ReinerSCT by modifying an external chip card reader with a display. For security reasons the computer was not equipped with a keyboard nor connected to any network during the election. During the voting process, the monitor of the voting machine as well as the trusted random number generator together with the input devices were separated by a screen, but the computer itself was visible. This made connecting anything to the computer without being detected by poll workers was impossible. The only input devices for the voter were the mouse and the chip card reader, the output devices were the monitor, the printer and the display of the trusted random number generator (the modified chip card reader).

To generate random numbers, the random number generator of a certified signature smart card from Siemens was used. The signature card was fixed and sealed inside a smart card reader with display and keypad. ReinerSCT had inserted the signature card and removed the protruding part of the card to prevent easy removal. The firmware of this special smart card reader was modified to request a random number from the random number generator of the signature card, display the random number and send it to the computer. The display was limited and consisted of two lines with 23 characters each. As the election required displaying more than two random numbers the arrow keys of the keypad were used to scroll through all random numbers. The smart card reader was also programmed to blank the display after 20 seconds without user action to prevent a voter from observing the trusted random number generator displaying the random numbers of a previous voter.

To unlock the voting machine, the voting authority handed out simple chip cards with small persistent memory on which the information about the eligibility of a voter was saved. The chip cards did not contain the name of the voter but instead contained a unique ID that was not associated with the voter. Each student was eligible for at least two and up to four out of fifteen different elections. Having a poll worker unlock the voting machine for each voter would have been error-prone and unnecessarily time-consuming. Using memory cards helped us enabling voters to do different elections on different days because the chip cards carried the information in which races a voter had voted after the voter had finished their voting process and returned the chip card. The voting authority already possessed a software and database infrastructure for the voters and the elections they were eligible for and used this not only for the Bingo Voting election but for the complete election. For each voter the voting authority saved which races they had cast a vote in the database. This allowed voters to cast votes for different races on different days and at different polling stations.

Using chip cards to unlock the Bingo Voting voting machine required additional security considerations. The voting authority had to prevent voters from forging valid chip cards, voting twice with the same chip card or cloning valid chip cards to vote multiple times. To prevent voters from voting with (copies of) the same chip card at different voting machines, each chip card was assigned to exactly one voting machine. This decision was made before it was decided to use only

one computer as voting machine during the voting process and keep a second computer as backup ready. Each chip card got a random number as a unique ID which was recorded and accepted only once by the voting machine. This prevented voters from voting multiple times by cloning a chip card. To prevent forgery and to ensure the chip card's integrity, the contents of the chip cards were digitally signed with every write access.

The voter's eligibility information was available on the chip card as one flag for each race. Each time a voter finished voting for a race, a *completion flag* was set on the chip card indicating this. The chip card reader used did not prevent the voter from removing the chip card at any time during the voting process. Because of this, a voter was able to remove the card after starting the voting process for a race but before the completion flag was set. Therefore each chip card contained an additional status bit for each race denoting if a voter had started voting for a race. This also enabled the poll workers to reconstruct during which part of the voting process the card was removed. If the status bits were in a wrong state during the voting process, the voting machine was locked.

4.2 Experiences

This section describes the experiences of the student parliament election. Section 4.2.1 describes the details of the preparation phase, Section 4.2.2 outlines the observations made during the voting phase and Section 4.2.3 describes the details of the tally phase. Section 4.2.4 summarises the observations and conclusions from the student parliament election.

4.2.1 Preparation Phase

The set-up of the election was done several days before the voting phase started on computers owned by the university and the dummy votes were later moved to the voting machine.

For the student parliament 72 candidates were nominated and each voter had nine votes to cast, with a maximum of five votes per candidate. So for this race we had to create $(5 \cdot 72 + 9 \cdot 2 =)$ 378 dummy votes for each eligible voter (including the additional dummy votes for the *abstention* and *invalid ballot* candidates, cf. Section 5.1.1).

The other races were a bit smaller with two to ten candidates and between one and ten votes per voter. We considered about 4000 eligible voters for the election of the student parliament and 1000 to 2000 eligible voters for each student body and the women's and foreign student's representative. These numbers were generous estimations for the number of voters of an election running for five days assuming that there were voters constantly voting each day from the opening of the polling station at 9 am till closing time at 6 pm. Generating the appropriate number of dummy votes took about ten hours on two servers with twelve AMD dual core Opterons with 1.8 GHz CPU clock rate each.

Since the elections were independent from each other, it was possible to do their set-up computations in parallel. Furthermore, the set-up computations for the student parliament's election were split into several parts without considerable loss of security. With optimal parallelization (i. e. by using more computers) the whole preparation phase probably could have been done within less than two hours.

For the proofs of correctness the election authorities tossed several coins and the outcome was used as challenge. Using Fiat-Shamir heuristics was impractical as the proof for the largest race was (4 000 voters · 378 dummy votes per voter · 1024 bits per commitment · 3 for the original commitments and the two new sets of commitments) approximately 580 MB for one instance just for the proof of equal distribution of dummy votes.

4.2.2 Voting Phase

Before entering the polling booth, the voter's eligibility was checked by a poll worker using the electoral register. This procedure was identical for the election with paper ballots and for the Bingo Voting election. When the eligibility was confirmed, the poll worker coded a chip card with a fresh ID and the information for which elections the voter was eligible and handed it to the voter.

Inside the voting booth the voter inserted the chip card into the chip card reader. The voting machine accepted a chip card if and only if the signature on the chip card was valid, its ID had not been used yet, its status bits were in a valid initial state and it contained the right machine name. If one of these four conditions was not fulfilled, the voting machine displayed a message telling the voter that the chip card was invalid. In that case the voting machine returned to a normal state after a few seconds, waiting for a valid chip card to be inserted. The voting machine only locked if an error occurred during a voting process.

After registering successfully at the voting machine and unlocking it, the voter chose with which race to start. The computer then displayed the corresponding ballot where the voter could distribute their votes. When the voter was satisfied with their choice, they clicked on a button to continue. Remaining votes were automatically given to the 'abstention' candidate. The 'invalid' candidate either got all or no votes since a ballot is either invalid or not. The voter was then presented a confirmation screen displaying their current ballot in order for them to double-check their distribution of votes. Then they could either return to the previous screen to change their choice, or confirm, and therewith cast, the ballot.

With the confirmation the votes were counted and the status bits on the chip card were set to a *race completed* state. Then the random number generator generated one fresh random number for each vote cast. The receipt was generated and printed and the voter was asked to compare the corresponding random numbers on their receipt with those on the random number generator's display. To support the voter a scheme of the receipt was displayed, indicating the positions of the fresh random numbers on the receipt. After checking the correctness of their receipt, the voter could go on with another race or end the voting process. The display of the random number generator was cleared automatically after a preset time, preventing the subsequent voter from learning the fresh random number(s) encoding the vote(s) of the previous voter.

Voters were allowed to vote in different elections at different days and at different polling stations. Therefore the poll workers noted in the electoral register in which races each voter had already cast their votes. This had to be done after each voter's voting process since the voter was allowed to decide to postpone their voting for certain races inside the polling booth. After leaving the polling booth the voter returned their chip card to the poll workers who checked the completion flag for each race and entered the elections in which the voter had cast their votes into

the electoral register.

4.2.3 Tally Phase

After the election the tally for each ballot box was calculated and published separately. As the voting machine was treated as one ballot box among others, it was unproblematic to tally and publish the tally of those votes cast at the Bingo Voting voting machine. In a different scenario, in which the ballot boxes were thrown together and tallied all at once, it would have been impossible to include the results of the Bingo Voting prototype in the total tally due to the cryptographic proofs giving the tally just for the voting machine.

The computing time for the tally was about five minutes. For the post-election proofs again the Fiat-Shamir heuristic was too cumbersome and we chose the necessary challenges in cooperation with the election authorities. The challenge was taken as a certain bit of the hash of a public document. As only a single bit was chosen, this method did not actually increase security. The computing time for the post-election proofs was about three hours on the same computer used for the pre-election proofs and successfully showed the correctness of the election.

4.2.4 Conclusions from the Student Parliament Election

The application of Bingo Voting in the election of the student parliament showed us that it is not sufficient to design a voting scheme that is good in theory. When used in practice, several additional difficulties come into play.

This early prototype used a mouse as user input device which was familiar to probably all voters. The computer affinity of the voters alleviated this problem but several mentioned that a touchscreen would have been preferred.

The main complaint of several voters was that the voting process took up to fifteen minutes. In addition to the complex election and the unusual user interface, several factors contributed to this. The main problem of the voting machine was a rather slow receipt printer that took several seconds to warm up and print a single page. This delay was perceived as too long after a long-winded voting process so that probably very few voters actually checked their receipts. The second problem contributing to the length of the voting process was the number of necessary write accesses to the chip card and the slow chip card reader.

The length of the voting process together with the size of the receipt, the largest using up two sides of a single sheet of paper and containing 378 random numbers, deterred many voters from checking the correctness of their receipt. This is acceptable since it is sufficient for the correctness of the voting system that only some voters verify their receipt, as long as the voting machine is unable to anticipate who is going to do so (see Section 6.2).

Some voters removed their chip card after the vote was cast but before the ‘finished’ flag was written, so it had an erroneous state. In all those cases the poll workers were able to retrace what happened without compromising secrecy or correctness of the election. Using a chip card reader that locks in the chip card would have solved this problem. It also would have sped up the voting process because most write accesses for changing the status bits could have been omitted.

Many voters expressed their missing trust in the random number generator used. To them the random number generator was just a black box. The fact that it was recognized as a modified chip card reader did not worsened the impression.

The conclusion of the experience with the student parliament election is that Bingo Voting is capable of handling elections up to a certain size and complexity without considerable loss of usability. For the acceptance, however, there is a big difference between being able to handle it and being able to vote comfortably. Several students decided to use the voting machine even though they could have used paper ballots, and most of them had no problems to cast their ballots. However, for a more representative evaluation of the usability and acceptance of the scheme, further studies should be made with a broader target group, including mature voters and people who are not used to using a computer.

The main problem limiting the size of elections for which the employment of Bingo Voting is practical is the size of the published data. Section 5 describes several methods to reduce the size of the published data. The limiting factors in this election on how fast a single voter could cast their vote were the chip card reader and the many read and write operations necessary for the reasons described above. The second problem was that a consumer-grade laser printer was used that took quite long to print a single receipt. The largest receipt used a complete DIN A4 page (two-sided). While it was still practical to check this receipt, it was not convenient.

4.3 Latest Prototype

For presentations and demonstrations the prototype for the student parliament election (see Section 4.1.4) is unsuited. Especially the activation with a chip card was unnecessary as it was no longer required to restrict each voter to only one voting process. This eliminated one big factor for the slow performance of the first prototype. The second reason for long waiting times, the printer, has been replaced as well. Instead of a laser printer with A4 paper, a cash register printer with thermal paper is used. It is much faster and the receipts are smaller. This would have been a problem for an election as large as the student parliament election but for a small mock-up election with only two votes per voter this is perfect.

To improve usability in addition to the reduction of the waiting time we have abandoned the mouse as input device and use a computer with a touchscreen instead. The latest prototype runs on a Beetle/iPOS system by Wincor Nixdorf. This computer was designed as a point of sale system and runs on a SuSE Linux. This prototype was successfully used to present Bingo Voting at the CeBIT 2009 and Messe Hannover 2009.

5. Improvements for Bingo Voting

The original description of Bingo Voting [BMQR07a, BMQR07b] provides a full-fledged protocol for a cryptographic voting scheme. The preparations for the student parliament election (described in Chapter 4) revealed several missing details and features that are often omitted in the description of a theoretical protocol but are essential for a real-world election. The first real-world application of Bingo Voting also made several extensions to the original protocol necessary to comply with the formal and practical requirements of the election. The election has shown that Bingo Voting is able to handle complex elections but also made clear that limitations exist due to usability and receipt size.

This chapter presents several extensions and improvements to the original Bingo Voting protocol. Section 5.1 presents several extensions necessary to use Bingo Voting in complex elections and points out possibilities and limitations of the voting scheme. Section 5.3 describes a presentation of random numbers and the limitations that usability requirements impose on the length of random numbers. While shorter random numbers make the verification process easier and thereby improve usability, they decrease the security as described in Section 6.3.5. Section 5.2 explains how changes of the voting scheme alleviates the problem of random numbers being drawn more than once during an election. Section 5.4 discusses the dispute resolution procedure of Bingo Voting. Section 5.5 discusses several ways to reduce the size of Bingo Voting proofs that are the biggest part of the data published (cf. Section 6.4.2). The hash chain presented in Section 5.6 tackles a problem that actually most cryptographic voting schemes possess: If an adversary gets hold of a receipt, the assumption that he does not know which votes are verified is violated.

5.1 Bingo Voting for Complex Elections

The original description of Bingo Voting in [BMQR07a] as well as the description in Chapter 3 present a Bingo Voting scheme for a simple election with only one vote per voter and a single voting machine. This section details the necessary changes to the basic Bingo Voting scheme for a complex election with more than one vote per voter and several voting machines.

Before discussing the details for complex elections, we look at the special cases of abstentions and invalid votes. The special proofs for the verification process of Bingo Voting require that each receipt and therefore each ballot contains the same number of marks (represented by fresh random numbers). This makes it impossible to accept a ballot with no or too few marks (which would be an abstention or a partial abstention respectively in an election with traditional paper ballots), or too many marks (which would be an invalid ballot). For this reason, Bingo Voting has to introduce the choice of abstention and invalid votes as additional candidates as described in Section 5.1.1.

Section 5.1.2 gives the details of the required changes to accommodate for more than one vote per voter. Section 5.1.3 argues why Bingo Voting is not suited for an election using single transferable vote (STV). Section 5.1.4 describes how to run a large election with several voting machines using Bingo Voting.

5.1.1 Abstention and Invalid Votes

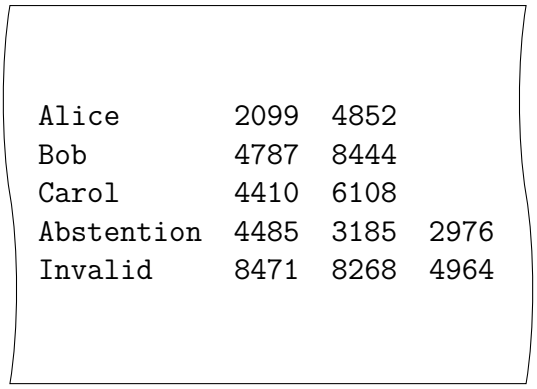
The changes described in this section were developed for the election described in Chapter 4 and were published in [BHMQ+08].

For the description of the integration of abstention and invalid votes we will consider an election with a single vote per voter. It is easy to expand this for more votes per voter using the method described in Section 5.1.2.

As mentioned above, Bingo Voting does not allow a voter to cast a vote with less or more choices marked than intended by the electoral system. The reason for this is the form of the proofs that each receipt is well-formed and contains the correct number of dummy random numbers (cf. Section 3.4.2). For the proof of each receipt, the voting authority makes a proof of a shuffle with known content. The input of this proof are the dummy votes used for the receipt together with the new commitment for the fresh random number representing the voter's choice together with a commitment to the corresponding candidate. If there was no fresh random number on the receipt, the proof would require only dummy votes as input, immediately marking the ballot as an abstention and therefore breaking voter privacy. If the receipt contained more than one fresh random number, the proof would have more fresh commitments as input accordingly. This would not only break voter privacy but force the voting authority to open the dummy votes used on the receipt in order to correctly and verifiably count the corresponding ballot as invalid, since some candidates did not lose dummy votes for the receipt despite not being elected (since the ballot was invalid).

The method chosen for the election described in Chapter 4 was to integrate abstention and invalid votes as separate candidates. This method makes handling those particular choices easy as they are treated almost like all other candidates in the verification process. It also accounts for the goal of electronic voting machines that casting an invalid ballot should be a deliberate choice and not an accident. This method is not specific for Bingo Voting and usable for many other cryptographic voting schemes. The method is necessary for any cryptographic voting scheme that require well-formed ballots to be cast by the voter.

Treating 'abstention' and 'invalid' as additional candidates works well with the extension described in Section 5.1.2. It allows for partial and full abstention by giving some or all votes to the 'abstention' candidate. There is only one small problem: In a normal election the ballot is either invalid or not but never partially



Alice	2099	4852	
Bob	4787	8444	
Carol	4410	6108	
Abstention	4485	3185	2976
Invalid	8471	8268	4964

Figure 5.1: Receipt for an election with $n = 3$ candidates, $s = 3$ votes per voter and a maximum of $v_{\max} = 2$ votes that each voter is allowed to cast for one candidate. This election allows the voter to abstain or cast an invalid vote using the method described in Section 5.1.1.

invalid. This means that either all votes are given to the 'invalid' candidate, or none. Unfortunately the proof that each receipt is well-formed does currently not check this. The correct distribution of votes to the 'invalid' candidate must therefore be asserted by the voting machine. This means that a corrupted voting machine may allow a voter to vote partially invalid by distributing only some votes to the 'invalid' candidate. This problem is discussed in Section 6.3.

5.1.2 Multiple Votes per Voter

The changes described in this section were developed for the election described in Chapter 4 and were published in [BHMQ+08].

The description of Bingo Voting in Chapter 3 is for a simple election with only one vote per voter. While this type of election is actually quite common, there are many elections that are more complex. In Germany, smaller political elections often allow the voter to make more in-depth decisions than just choosing one candidate. The election described in Chapter 4 is a good example for a more complex election. As the original publication [BMQR07a] states, it is actually rather straightforward how to expand Bingo Voting to allow for more complex elections.

In this section, we consider an election with n candidates plus invalid and abstention (cf. Section 5.1.1) in which each voter has s votes and may distribute them among the candidates up to a maximum of $v_{\max}$ votes per candidate.

In the basic scheme, a Bingo Voting receipt contains one random number associated with one candidate, representing a possible choice of the voter. During the verification process there is a proof for each receipt that all random numbers but one are dummy random numbers (see Section 3.4). In an election in which each voter has more than one vote and may give a candidate more than one, several changes to this system are necessary.

It is not very complicated to model an election with multiple votes per voter. The basic idea is to give each voter one receipt for each vote the voter casts. But instead of printing s distinct receipts those are combined onto a single receipt by simply printing additional random numbers next to each candidate name. Fig-

ure 5.1 shows an example of a receipt of an election with three candidates, in which each voter has three votes and is allowed to cast up to two votes for a single candidate.

In an election with n candidates, s votes per voter and a maximum of $v_{\max}$ votes per candidate each receipt contains $v_{\max}$ random numbers next to each candidate name plus s random numbers next to the 'abstention' and 'invalid' candidate. The reason for this simply is that not more than $v_{\max}$ fresh random numbers (corresponding to actual votes by the voter) may be given to any candidate. The 'abstention' and 'invalid' candidate are special in this regard, as a voter may choose to abstain completely, so all their votes are counted as cast for 'abstention'. The 'invalid' candidate is similar except that a voter may only cast a valid ballot (with no votes given to the 'invalid' candidate) or cast an invalid ballot which contains only votes for the 'invalid' candidate.

This changes not only the receipts but consequently also the proof that each receipt is well formed. This change, too, is straightforward: The proof has the set of dummy votes and commitments for the fresh random numbers as input and shows that the content of these commitments accord with the receipt. It is important to note, however, that the proof does not show that either none or all votes are cast for the 'invalid' candidate. See Section 6.3 for a discussion of this.

5.1.3 Ranking Candidates

An election using instant run-off, single transferable vote (STV) or any other voting system that asks the voter to rank candidates (see Section 2.1.1) requires a voting scheme that accounts for ballots with votes of different qualities ("ranked votes") that are linked in the tally. When a candidate is eliminated for having the fewest votes during the tally for an instant run-off election, the voting authority must be able to distribute the corresponding ballots according to the second vote on the ballot.

This differs in two ways from the principles of Bingo Voting: In a Bingo Voting election there is only one type of vote, and a Bingo Voting tally only publishes the sum of all votes for each candidate and no further information about single ballots. This increases voter privacy and makes pattern voting impossible (see Section 6.3.1), but unfortunately makes ranking candidates impossible with an unmodified Bingo Voting scheme.

There is a way, however, to change the Bingo Voting protocol so that it is possible to reconstruct ballots. For this the dummy votes are expanded so that each not only contains a commitment to a candidate and one to a dummy random number but also a commitment to a ballot id. The dummy votes used for one ballot are also published together, similar to the suggestion of prearranged dummy votes described in Section 5.2. In addition to that, the voting authority prepares dummy votes for n^2 candidates, instead of the n original candidates of the election. These candidates are called $\texttt{Cand}_i|\mathit{rank}$, where $\texttt{Cand}_i$ is the ith candidate and *rank* goes from 1 to n for each candidate.

The changes to the proofs are as follows. Before the election, the voting authority proves that for each ballot there is exactly one dummy vote for each combination of candidate and rank, and that all have the same ballot id. This proof must employ randomized partial checking, as this allows the voting authority to prove that the shuffle keeps all dummy votes of a ballot together. These proofs show

only the content of the commitments to the candidates and the ballot ids but not the content of the commitments containing the dummy random numbers. The generation of the receipt is similar to the receipt generation process in an election using sliced clusters (cf. Figure 5.2). The positions of the fresh random numbers on the receipt give the ranks the voter assigned to the candidates. After the election phase, the voting authority opens all unused dummy votes using a shuffle of known content. This may be done using the more efficient proof style described in Section 5.5.2. Since the dummy votes all carry a distinct ballot id, the voting authority is able to reconstruct each ballot. The voting authority also proofs for each ballot that it is well-formed. This proof opens the dummy votes used for the receipt together with new commitments for the fresh random numbers, but does not open the parts of the dummy votes containing the ballot ids. These proofs may also use the more efficient proof style described in Section 5.5.2.

The receipts for an election with ranking become very large as there are n^2 random numbers on a receipt for an election with n candidates. Also the public data is larger since several of the improvements presented in Section 5.5 are not applicable. Nevertheless, the changes to Bingo Voting presented here make an election with Bingo Voting that require the voter to rank the candidates possible.

5.1.4 Multiple Voting Machine

The presentation of Bingo Voting in Chapter 3 assumes that there is only a single voting machine used for the election. For most elections, a single voting machine is not sufficient to conduct an election.

The intuitive way to expand Bingo Voting for use in larger elections is to treat each voting machine as an individual election and later simply add the tallies. This is actually not different from the method used by many real-world elections using paper ballots. In many countries, for example in Germany, each ballot box is tallied separately and later the tallies are added for the final tally of the complete election.

This may pose a problem when only very few people cast their vote into a specific ballot box. For this reason some elections, for example elections in Ireland, require a *centralized tally*. This means that ballot boxes are not tallied separately but all ballot boxes are taken to a *central counting place*, and the ballots of all ballot boxes are mixed and tallied as a whole. This means that there are no intermediate results for single ballot boxes. The goal is to leak as little information as possible about single votes, even when a ballot box contains only a few or even a single vote.

It is possible to obtain a similar level of voter privacy with Bingo Voting. There are two ways to accomplish a centralized tally in an election using Bingo Voting. The first way is that instead of treating each voting machine as a single election the voting authority only prepares a single set of dummy votes and distributes this among all voting machines used while keeping the information which dummy vote was allocated to which voting machine secret. The final proof then contains no more information about the intermediate results of single voting machines than the final tally.

The second way is using the method of prearranging dummy votes as described in Section 5.2 in order to not only decrease the probability of collisions but also increase voter privacy. This methods requires that unused dummy votes are not opened directly for the tally but instead they are opened using a proof of a shuffle

with known content (cf. Section 2.4.4). If all unused dummy votes of an election are opened this way together without showing which candidate originates from which dummy vote, this only divulges the final tally and not any intermediate results.

5.2 Reducing the Length of Random Numbers

Bingo Voting uses random numbers to represent potential and actual votes in an election and on a receipt. The security of Bingo Voting is based on the assumption that a random number is not drawn more than once within an election. If this happens, we call this a *collision*. The original description of Bingo Voting [BMQR07a] assumes that the random numbers are long enough such that no collisions occur or that at least their number is small enough to be ignored. If a collision occurs, however, a corrupted voting machine may change a single vote. Section 6.3.5 discusses the consequences of collisions in detail.

The length of the random numbers used in a Bingo Voting election has to be chosen so that collisions are sufficiently improbable. This is easily done by choosing very long random numbers. This poses another problem, however. The only time a voter is confronted with details specific to Bingo Voting during the voting process is when checking the correctness of the receipt by comparing the random numbers (cf. Section 3.3.2 and Section 6.4.1). This leads to a second, conflicting requirement: Random numbers must not only be long enough to avoid collisions, their comparison has to be manageable by every voter as well. Even though the comparison is voluntary it should be made as convenient as possible for the voter in order to increase acceptance and, by making the comparison more likely, also security. This makes the length as well as the presentation of random numbers in Bingo Voting important. The presentation of random numbers is discussed in Section 5.3.

This section describes a change to the original Bingo Voting protocol that makes the occurrence of collisions that allow a corrupted voting machine to change a vote less likely. Prearranging dummy votes and determining beforehand which dummy votes are to be used for which receipt prevent a voting machine from capitalizing on a collision occurring during the voting phase.

The problem of a collision during the voting phases arises when the freshly drawn random number is identical to a dummy random number for the elected candidate that is valid for use on the receipt for the current voter. The original Bingo Voting protocol does not specify how the voting machine chooses the dummy votes that are to be used for a receipt. That means that a corrupted voting machine is able to choose from all dummy votes that were not used before. At the beginning of the voting phase, the voting machine is able to choose from all dummy votes for a candidate, making the risk of being cheated by a corrupted voting machine larger at the beginning of the voting phase than at the end. It also means that if the voting authority prepares more dummy votes this risk is increased further.

The consequence of this observation is to limit the voting machine's choice by prearranging dummy votes in *clusters* and determining the order in which the clusters are to be used during the preparation phase. Each cluster consists of as many dummy votes for each candidate as there are random numbers on a receipt

for this candidate. In a simple election with one vote per voter this means one dummy vote for each candidate.

The simplest method is that the voting authority publishes the clusters of dummy votes in the same order in which the voting machines uses them for the receipts. This is best complemented by the use of hash chains (see Section 5.6) that also make the order in which the voting machine issues receipts apparent. This way an auditor is able to verify that the clusters were used in the correct order determined by the voting authority before the election. For this the auditor simply has to check that the proof that the receipt is well-formed uses the dummy votes of the corresponding cluster (minus the unused dummy votes).

The effect of this method is that the voting machine is able to change a vote only if the fresh random number is identical to the dummy random number of the elected candidate that is in the cluster the voting machine has to use for the receipt of this specific voter. This reduces the chance of such a collision occurring drastically and allows for shorter random numbers. Section 6.4.1 analyses and compares the length of random numbers required for an election using Bingo Voting with and without prearranged dummy votes.

The main disadvantage of prearranging dummy votes is that voter privacy is broken if unused dummy votes are simply opened after the election. Since it is obvious which cluster an unused dummy vote belongs to (since this is published and required to verify that the voting machine used the correct cluster), the unused dummy votes of a cluster directly give the voter's choices. To prevent this the voting authority has to open unused dummy votes using a shuffle of known content (see Section 2.4.4) to show the content of these dummy votes without showing which dummy vote has which content. This increases the size of the public data, however.

Another disadvantage is that for the verification process of the correct order of use of the clusters the order in which the receipts were issued becomes apparent. This may or may not be acceptable for an election depending on the requirements. A solution similar to the propositions for hash chains that do not reveal the order of the receipts (see Section 5.6.3) seems possible, but currently there is no durable suggestion for this.

For complex elections in which a voter has more than one vote there are two different variants of how to prearrange dummy votes and implement clusters. The first variant uses clusters as described above with one dummy vote for each random number on the receipt. The voting machine is able to freely arrange the random numbers on the receipt (as long as each dummy vote is correctly associated to its candidate). This means that fresh random numbers may be written first next to a candidate, followed by dummy random numbers if applicable. This way of arranging random numbers makes it easier for the voter to find the fresh random numbers for the comparison during the first verification step in the voting booth. But at the same time the chance of a collision is increased since there is not only one but several dummy random numbers a fresh random number may be identical to. To prevent this, the voting authority may specify the order in which the dummy votes within the cluster are to be used. For this, the voting authority further divides the cluster into slices, each slice containing one dummy vote for each candidate. This variant is called *sliced clustering* in contrast to the first variant that is called *unsliced clustering*. The voting machine has to use the slices

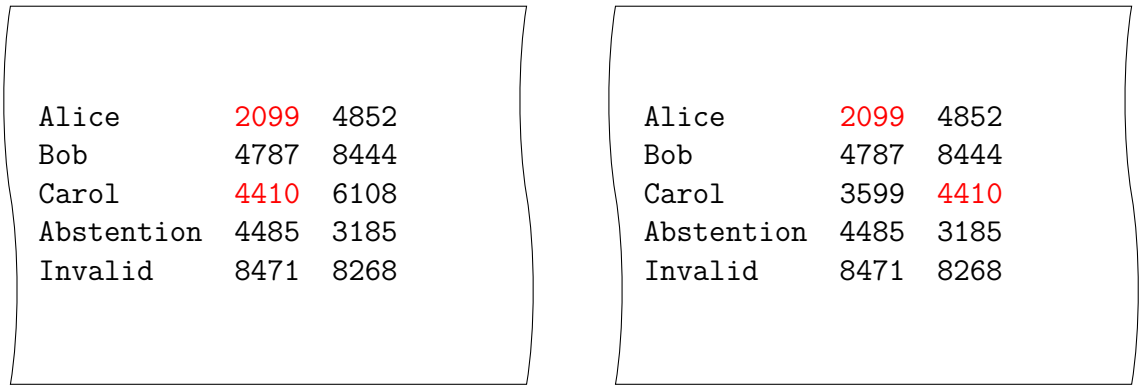


Figure 5.2: A comparison of Bingo Voting receipts in an election with three candidates and two votes per voter. Fresh random numbers are indicated in red. The receipt on the left is from an election using unsliced clusters, so fresh random numbers are written first next to a candidate, dummy random numbers afterwards. The receipt on the right is from an election using sliced clusters so the first random number is written in the first column, the second random number in the second column.

A	C	E	F	G	H	J	K	L	M	N	P	Q	R	S	T
U	V	W	X	Y	Z	1	2	3	4	5	6	7	8	9	0

Figure 5.3: Alphabet with 32 easily distinguishable characters for the presentation of random numbers

in the correct order, for each slice only one fresh random number is to be used (otherwise information about the voter's choice, namely how many votes were distributed to the same candidate, is leaked). The consequence is that on the receipt each column of random numbers (corresponding to the slices) contains one random number, making finding fresh random numbers on the receipt for the individual verification process in the voting booth more cumbersome. Sliced clustering is also unsuitable for elections in which voters have more votes than they are allowed to cumulate on a single candidate. The reason is that a cluster for such an election would contain slices containing only the dummy votes for the two special candidates 'invalid' and 'abstention'.

5.3 Presentation of Random Numbers

In an election employing Bingo Voting, the first step in the individual verification process happens during the voting process inside the voting booth (cf. Section 3.4.1). After having cast their vote, a voter receives a receipt and is able to check that it correctly encodes their choice. For this, the voter has to compare two strings and reliably decide whether or not both are identical. One string is displayed by the trusted random number generator, the other is printed on the receipt next to the elected candidate. It is desirable to make this comparison as easy and convenient as possible. It not only increases the chance with which a manipulation attempt is caught and thereby increases security but also helps with

acceptance.

Making the random numbers shorter makes the comparison simpler but also increases the probability with which a collision occurs. Section 6.3.5 gives a detailed explanation of how a corrupted voting machine is able to change a vote when a collision occurs. Section 5.2 describes how to change Bingo Voting in order to alleviate the problem of collisions. In a practical employment Bingo Voting will have to use random numbers long enough to make collisions sufficiently improbable. In order to improve usability despite this constraint determined by the number of dummy votes in total we now look at the presentation of the random numbers.

Using numbers and uppercase letters from the English alphabet we have 36 different familiar characters. Some of these are too similar to be reliably distinguished, the most prominent example being the uppercase letter O and the number 0. Eliminating those characters we obtain a set of 32 distinguishable characters as presented in Figure 5.3.

As implied by [Rya69] and [Wic64], humans process strings more easily if they are grouped into chunks of three characters each. Experiments showed that grouping into chunks of three digits is considerably superior to groups of two digits and a little superior to groups of four or more digits. To aid the comparison process, characters should be printed in groups of three with a hyphen separating the groups.

With 32 different characters, each group of three characters represents 15 bit of information.

5.4 Dispute Resolution

The goal of a cryptographic voting scheme is for the voter to be able to verify that their vote was correctly included in the tally. If the individual verification fails, the voter knows that this was not the case and a dispute arises. This section discusses the dispute resolution process of Bingo Voting and proposes several solutions to make Bingo Voting dispute-free. The results presented here are published in [BHK+09].

The dispute resolution procedure should ensure two things. A voter whose vote was manipulated by the adversary, either through a corrupted voting machine or by a corrupted voting authority, should be able to prove that some manipulation took place and their vote was altered. At the same time the adversary should not be able to claim that the election was manipulated even if this is not the case. Otherwise it would be possible for the adversary to abort an election without a result by claiming that votes were altered even if the tally was correct. This would be disastrous for a real-world election.

One goal of Bingo Voting is to allow voters to not only detect a manipulation of their vote but also prove the occurrence of such manipulation should the situation arise. During the voting process, after the voter has cast their vote they receive a receipt encoding their vote. The voter is able to verify this inside the voting booth as described in Section 3.4.1. During the tally phase copies of all receipts are published and the voter is therefore able to verify the correct inclusion of their vote in the tally. If at one point this verification fails, it is desirable that the voter is able to prove this and contest the election. In an ideal case the voter is able to do this without revealing their vote but unfortunately this is not always

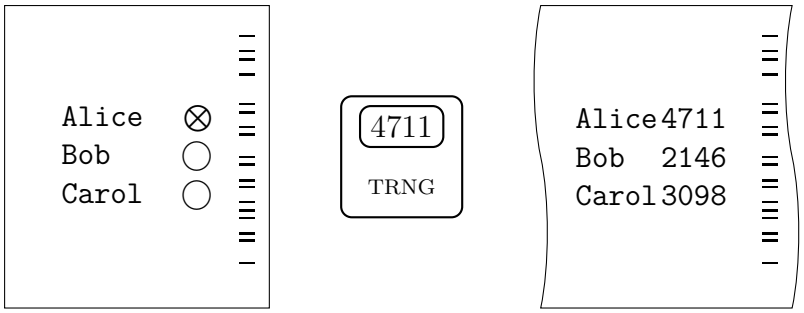


Figure 5.4: To create forensic evidence the voter enters their choice using a paper ballot (left). The voter then checks if the random number of the trusted random number generator (middle) is placed correctly on the receipt (right). If the receipt is not correct, the alignment information printed onto the border of the ballot and the receipt enable the use of privacy sleeves for dispute resolution without disclosing the voter's choice.

possible. Sometimes it is considered sufficient to prove this while revealing the voter's choice only to a trusted arbiter.

Section 5.4.1 looks at the dispute resolution procedure during the voting phase, Section 5.4.2 discusses the voting procedure during the tally phase.

5.4.1 Dispute Resolution During the Voting Phase

A dispute arises during the voting phase if a voter notices in the voting booth that their receipt is incorrect. In this case the voter should be able to prove that the receipt does not represent their choice. An advantage of Bingo Voting is that it works with an arbitrary user interface (cf.Section 6.4.1). This makes it possible to add a verification mechanism to an arbitrary voting machine.

If a voter enters their choice using a touchscreen interface, there is evidence missing if a corrupted voting machine produces an incorrect receipt. The machine simply acts as if the voter had chosen a different party. Even if the voter notices the discrepancy and alerts a representative of the voting authority to resolve the dispute, the situation is indistinguishable from the voter having chosen this different party in the first place.

A possible solution is to combine Bingo Voting with a scanner-based interface. The voter first records their choice on paper that is then fed into the scanner. The scanner keeps the paper ballot inside a special compartment until the voter is convinced that the receipt is correct. In addition the display of the trusted random number generator only is cleared under the same condition. This way there is enough forensic evidence for a voter to prove that the receipt is incorrect and does not encode the choice made on the paper ballot.

Disclosing this evidence in a naïve way, however, would violate ballot secrecy. While it is possible to alleviate this problem by including a trusted arbiter this may still be considered less than ideal.

Instead of involving a trusted third party, a technical solution similar to Scantegrity II [CCC+08] seems promising. For this, the scanner prints a random bar code onto one margin of the paper ballot during the scanning process. This bar code is used as alignment information in case of a dispute, the receipt contains

an identical bar code so that the positions of candidates are identical on paper ballot and receipt relative to this bar code. In case of a dispute, the voter puts paper ballot and receipt inside privacy sleeves. There are two kinds of privacy sleeves that both leave the bar code uncovered so that it is possible to align and compare paper ballot and receipt during the dispute resolution process. The first type of privacy sleeve reveals the candidate names in addition to the alignment information and is used if the candidates are not placed identically in respect to the bar code on the receipt and the paper ballot. The second type covers the candidate list as well as everything but one row of random numbers on the receipt and the marking area for one candidate on the paper ballot respectively together with the corresponding part of the alignment information. The voter uses this second type of privacy sleeves to prove a discrepancy between their choice and the receipt if the alignment information is correct. Figure 5.5 shows both pairs of privacy sleeves.

In case of a discrepancy, there must be either a row with the fresh random number but without a mark of the voter or the alignment information on the ballot and on the receipt must differ. Both discrepancies can be proven by covering up the aligned ballot and receipt except for those places necessary to see the discrepancy. So the proof consists either of a row containing the fresh random number but no mark without revealing which row this is or the proof consists of the two differing alignment bar codes without showing the mark at all.

This scheme allows the voter to prove an error or manipulation without leaking information about the voter's choice. To prevent anyone from learning the content of the ballot the voter has to make the alignment without outside help. This problem with usability is likely to decrease its applicability and acceptance.

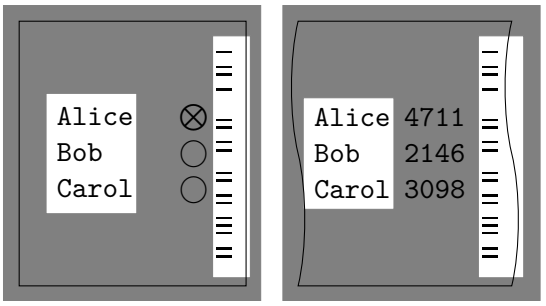
Finding an easy to use technical solution to prove an error or a manipulation in the voting booth remains an open problem.

5.4.2 Dispute Resolution During the Tally Phase

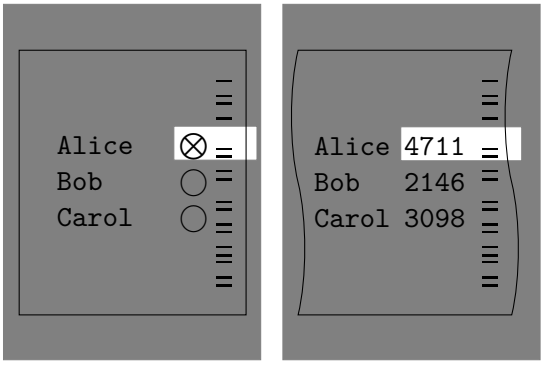
In the tally phase, the voting authority publishes a copy of each receipt. A voter is able to verify that their receipt was published correctly by comparing the published copy with their original. If they copy was altered or if their receipt is missing altogether, this indicates that something went wrong.

To successfully contest the election the voter must be able to prove that there is indeed a discrepancy between the receipt they were issued during the voting process and the digital copy published afterwards. Strictly speaking, possession of the original receipt is not sufficient for this unless the party responsible for dispute resolution is able to verify that the receipt was indeed issued by a voting machine during the election.

A simple approach is to include a digital signature of the voting machine on the receipt. This method was used for the election described in Chapter 4. While this prevents the voter from forging a receipt (under the assumption that the signature scheme is not broken) it assumes that the voting machine is uncorrupted and does not simply issue an invalid signature. To alleviate this problem it is possible to introduce an additional party that produces the signature, for example a signature card as part of the printer. This signature card would produce a digital signature for the data the printer receives from the voting machine, i. e. the receipt information, and the printer adds the digital signature to the printed receipt. This



5.5.1: privacy sleeves to check the correctness of the alignment information



5.5.2: privacy sleeves to check the correct encoding of the voter's choice

Figure 5.5: When a dispute arises during the voting process the voter may choose one of two pairs of privacy sleeves to protect the secrecy of their ballot. The first pair show the candidate names and alignment information but hides the voter's choice on the ballot as well as the random numbers on the receipt. The second pair displays one marking area and the corresponding alignment information on the ballot and one random number with the corresponding alignment information on the receipt. Note that the privacy sleeves used to check the correct encoding of the voter's choice must be larger than the paper ballot and the receipt respectively to avoid that the process leaks information about the position of the part shown (and therefore the voter's choice).

way the voter is able to contest the election if the printer containing the signature card is uncorrupted.

To solve the problem of contesting an election in which voting machine and printer are corrupted, the voting authority may fall back to using physical security, for example special unforgeable paper for the receipts. There are several approaches to unforgeable paper, many of which are used for bank notes. Since the authenticity of a receipt is only checked in case of a dispute, other methods that are impractical for paper bills may be used. One example is marking paper with DNA strands that may even be unique for each voting machine. While the verification procedure is time-consuming and requires a laboratory the application of a DNA solution is easy and the method is established.

Having two independent ways to prove a receipt to be authentic is a big advantage. If the printer were corrupted, the security would fall back to the security

provided by the special paper. If, however, the printer works correctly, contesting an election becomes easy as only the correctness of the signature must be verified.

One big disadvantage of all approaches described here is that the voter is not able to immediately verify that the receipt issued during the voting phase would later be recognized as an authentic receipt in case of a dispute. This remains an open problem.

The procedure to contesting an election during the tally phase described here requires the original receipt. The problem that a voter who no longer possesses their receipt is unable to challenge manipulations is discussed in Section 5.6.

5.5 Reducing the Size of Public Data

Like most cryptographic voting schemes, Bingo Voting requires the publication of data to allow auditors to verify the correctness of the result and to convince voters that their vote was correctly included in the tally. For Bingo Voting, the voting authority publishes the dummy votes before the voting phase (see Section 3.3.1). After the voting phase the voting authority publishes a copy of each receipt (see Section 3.3.3). In addition to this the voting authority produces cryptographic proofs for the correctness of the tally. These proofs essentially prove two things: Each candidate has received the same number of dummy votes before the election, and each receipt contains the correct number of dummy random numbers.

One large obstacle for the employment of Bingo Voting in real-world elections is the size of this public data. The data published during the election described in Section 4 was 580 MB for a single instance of the proof of equal distribution of dummy votes for a single race with 4000 voters, 72 candidates and 9 votes per voter. The original Bingo Voting scheme requires about 80 instances of these proofs in order to use the Fiat-Shamir heuristic to turn it into a non-interactive zero-knowledge proof. This would have turned the size of the public data well above 26 GB for a small, albeit fairly complex, election. The size of the public data increases linearly with the product of the number of voters, the number of candidates and the number of votes per voter. This makes it obvious why the size of the public data poses a problem for the practicality of Bingo Voting for large-scale elections with millions of voters. The main factors for the size of the public data are the size of each commitment and the size and numbers of the zero-knowledge proofs proposed in the original version.

This section describes several changes to the Bingo Voting protocol in order to reduce the size of the public data. Section 5.5.1 describes how the use of elliptic curves reduces the size of a single commitment. Section 5.5.2 suggests applying a different proof system, reducing the size of each zero-knowledge proof used for Bingo Voting. Section 5.5.3 explains a change to the proof that each candidate received the same number of dummy votes before the election, reducing the number of proofs by proving this during the tally phase.

To demonstrate the improvements of the changes described in this section, Section 6.4.2 presents the size of the public data for three different scenarios of real-world elections using Bingo Voting with the changes described in this section and compares it to Bingo Voting without these improvements.

5.5.1 Decreasing Dummy Vote Size

As the number of dummy votes is determined by each specific election, namely by how many candidates there are and how many votes each voter may distribute, as well as the number of voters, optimizing the size required for a single commitment is the best way to reduce the size of the published dummy votes.

Bingo Voting uses UHDLCs (see Section 3.1.2). In the original version [BMQR07a], the group $(G, \cdot)$ used was $\left(\mathbb{Z}_p^\times, \cdot\right)$, the multiplicative group of units of a group of integers modulo p with p prime. In order to be secure, p should be at least a 1024 bit prime number. As any commitment is a group element of G, this means that every commitment requires 1024 bit to be represented despite the content being much smaller.

Koblitz [Kob87] and Miller [Mil86] independently proposed the use of elliptic curves over a finite field for cryptographic purpose. They offer a smaller key size while providing a comparable security to finite fields.

Using a cyclic subgroup $\tilde{E}$ of prime order p of an elliptic curve E over $\mathbb{F}_{2^n}$ thus reduces the size of each commitment without reducing the security (namely the binding property, since the commitment scheme is unconditionally hiding). Lindell et al. [LPS08] suggest using an elliptic curve with $n \approx 256$. We continue to use the multiplicative notion to avoid confusion with previous descriptions of UHDLCs.

To use UHDLCs the voting authority must publicly choose two generators g and h of the subgroup (such that $\langle g \rangle = \langle h \rangle = \tilde{E}$). The generators g and h must be chosen such that no party knows the discrete logarithm of g with respect to h. Each commitment is then a point of E and can be represented with only $n + 1$ bits [Ser98]. With $n = 256$ this means that each commitment now only has a quarter of the size of the commitments of the original scheme.

5.5.2 Changing the Proof Style

The proofs published for the verification process of Bingo Voting (see Section 3.4) are all non-interactive zero-knowledge proofs of a shuffle with known content. The prover (i. e. the voting authority) publishes a set of commitments and a list of plaintexts, and proves that each plaintext is the content of exactly one commitment. This means the prover shows the content of each commitment without showing which commitment contains which plaintext.

In the original description of Bingo Voting the zero-knowledge proofs used were randomized partial checking (RPC) proofs (see Section 2.4.4.1 for detailed presentation of randomized partial checking). One disadvantage of this proof style is that it requires many instances to use the Fiat-Shamir heuristic to turn the proof into a non-interactive zero-knowledge proof and two rerandomizations per instance. Each instance of these proofs allow the adversary to cheat undetected with probability $\frac{1}{2}$. This makes k instances necessary to achieve a cheating probability of 2^{-k}, which leads to very large proofs. This section proposes employing a proof technique described by Groth [Gro02, Gro10] and using the multiplication proof by Damgård and Jurik [DJ01]. The proof technique was originally described for Paillier ciphertexts. The scheme used for Bingo Voting was adapted to UHDLCs.

Before it is possible to use the proof of a shuffle of known content as proposed by Groth, it is necessary to combine the pair of commitments that form a dummy vote into a single commitment. This is easily done in a way that the new commitment

verifiably contains the content of both commitments forming a dummy vote. Let

$$\nu = \lceil \log n \rceil$$

be the length of the representation of the n candidates and $(\textsc{Commit}(\mathtt{r}), \textsc{Commit}(\mathtt{Cand}))$ a commitment for candidate `Cand` with dummy random number `r`, then

$$\hat{c} = \textsc{Commit}(\mathtt{r})^{2^{\nu}} \cdot \textsc{Commit}(\mathtt{Cand}) \tag{5.1}$$

is a commitment to $(\mathtt{r}|\mathtt{Cand})$. This fact allows us to use the proof of a shuffle of known content proposed by Groth [Gro02, Gro10]. It also means that for opening a dummy vote it is sufficient to publish the randomness of the combined commitment $\hat{c}$. A similar method is used by Groth when turning a proof of a shuffle of known content into a proof of a shuffle of homomorphic encryptions. Without combining the two commitments of a dummy vote into a single commitment the use of the proof of a shuffle of known content would not be possible efficiently since the association between dummy random number and candidate would be destroyed. This association is fundamental to prove correctness of Bingo Voting.

This proof has a length of $8{\cdot}n{\cdot}\text{size}_{\text{COM}}$ bits where n is the number of commitments and plaintexts and size_{COM} is the size of a single commitment, a message or a random number used in the commitment. Compared to this the randomized partial checking protocol used in the original Bingo Voting scheme has a length of $n \cdot 5 \cdot \text{size}_{\text{COM}} \cdot 80$ bits (with the adversary having a chance of cheating undetected with probability 2^{-80}) which is significantly larger. Section 6.4.2 presents the total amount of the reduction of the size of public proofs for a Bingo Voting election.

5.5.3 Optimizing the Proof of Correct Distribution of Dummy Votes

During the preparation phase, the number of dummy votes generated for each candidate must be the same. If this is not the case, the number of unused dummy votes after the voting process (minus the number of nonvoters) do not give the tally.

In the original version of Bingo Voting the voting authority publishes the set of dummy votes together with a proof that each candidate has received the same number of dummy votes (see Section 3.3.1). The size of this proof depends on the number of dummy votes and therefore on the expected number of voters. The change to this proof of the correct distribution of dummy votes presented here makes the size of the proof depending on the number of voters that actually attended the election.

The central idea of the change proposed in this section is that is it unnecessary to include those dummy votes into the proof that are opened during the tally phase. Those dummy votes were not used on any receipt during the voting phase. It exploits the fact that it is actually sufficient to prove the correct distribution of dummy votes after the voting phase.

Instead of proving before the voting phase that the published dummy votes are equally distributed, i. e. that there is the same number of dummy votes for each candidate, the voting authority proves after the voting phase that the number of unused dummy votes plus the number of opened dummy votes for each candidate is the same.

More precisely, let d be the number of dummy votes published during the preparation phase for all n candidates, t the election turnout (with $0 \leq t \leq 1$), and $\hat{v}$ the number of votes cast. Candidate $\texttt{Cand}_i$ should have lost $d_i \cdot t - \hat{v}_i$ dummy votes, with $d_i = \frac{d}{n}$ being the number of dummy votes for $\texttt{Cand}_i$ and $\hat{v}_i$ being the number of votes cast for $\texttt{Cand}_i$. The voting authority can prove this after the election, decreasing the number of dummy votes included in the proof from d to $d \cdot t - c$ since only used dummy votes have to be included.

While the improvement of efficiency of this change itself is small, this change makes the proof size depending on the number of actuals voters (see Section 6.4.2). This allows the voting authority to generously prepare dummy votes without unnecessarily increasing the size of the public data. Unused dummy votes are still published during the preparation phase and are opened during the tally phase but are no longer part of a proof.

5.6 Preventing the Receipt Stealing Attack

Most cryptographic voting schemes are based on the assumption that each receipt is checked or at least that enough receipts are checked so that a significantly large manipulation is caught with high probability. The first assumption is very strong, and the second assumption either means all but very few receipts are checked (still a very strong assumption) or that the adversary does not know in advance which receipts are checked.

The receipt stealing attack described in Section 5.6.1 is a simple attack destroying this fundamental assumption. Section 5.6.2 proposes a method of linking receipts together by means of a hash chain. One drawback of this method is that the order in which receipts were issued by the voting machine becomes apparent. Section 5.6.3 describes a modification of the hash chain method that avoids this problem. Section 5.6.4 discusses the relevance of the receipt stealing attack and the hash chain method for other cryptographic voting schemes. The results presented in this section were published in [BHK+09].

5.6.1 Receipt Stealing Attack

Most cryptographic voting schemes offer a receipt to each voter to help them to verify the correct inclusion of their vote into the tally. To check this, voters compare their receipts to the data published by the voting authority.

The receipts of most voting schemes contain too much information for the voter to memorize the complete receipt. This is especially true for random strings (truly random or quasi-random such as hash values, signatures or serial numbers) that are common to cryptographic voting schemes. Scantegrity II (cf. Section 2.5.8) is probably the cryptographic voting scheme with the least information on the receipt. The receipt consists of a three letter code and the serial number of the ballot. While the three letter code can probably be memorized the serial number might be challenging depending on the length. This makes Scantegrity II one of the few (if not the only) cryptographic voting scheme in which a voter may be able to check the correct inclusion of their vote without a receipt.

For most other voting schemes the genuine receipt or an exact copy is required for verification. This means that once an adversary has got hold of a receipt he can be fairly sure that the corresponding vote will not be checked if no copy exists.

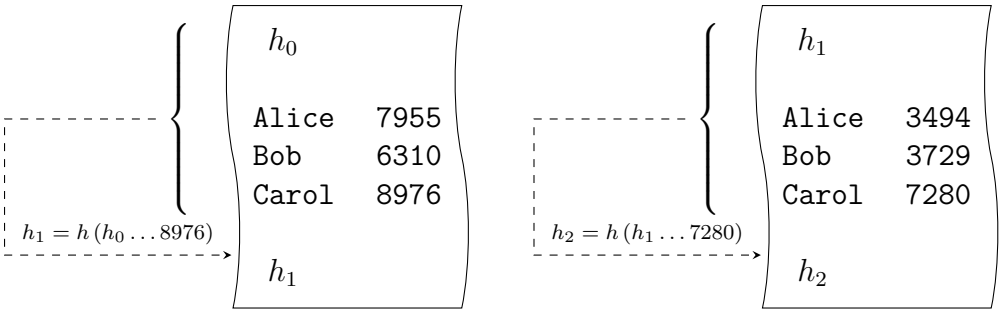


Figure 5.6: Two consecutive receipts are linked by incorporating the hash value of the previous receipt into the hash value of the current receipt. The so-formed hash chain is broken if only one receipt is altered.

There are several possibilities how an adversary can get hold of a receipt, some of them have been pointed out in [Rya06], [RS07], [RP05] and [Dan08]. The adversary can search trash bins for receipts directly in the polling station or in the vicinity (or put one up himself), steal them, buy them or collect them ensuring the voters that they will be checked. Independent of the exact way an adversary acquires the receipt we call this *receipt stealing*.

If an adversary has not only obtained a number of receipts but also has access to the voting machine, he can change the votes that belong to the obtained receipts. This is a problem that many cryptographic voting schemes share. Rivest and Smith call this a *collusive attack* [RS07], as the adversary obtaining the receipts must collude with the voting authority, or at least with a part of the voting authority that controls the voting machine or bulletin board. Ryan and Peacock [RP05] and Karlof, Sastry and Wagner [KSW05] also identified this problem and suggest voter education as a countermeasure. Ryan and Peacock also propose the introduction of a VVPAT style mechanism that produces two receipts, one for the voter and one remaining at the polling station. This allows auditors to verify each receipt using this copy, making the verification by the voter redundant.

The education of voters is important and may help against carelessly discarded receipts and make stealing receipts more difficult, but the success against receipt buying is doubtful. Making the voter's receipt redundant forces the adversary to obtain two receipts to ensure that a manipulation is not detected. But the knowledge that their receipt is redundant may make the voter less likely to check it. At the same time, this measure relies on auditors checking the copies of the receipts that remain at each polling station which requires a sufficient number of auditors.

The hash chains proposed in the next section make retaining a copy of each receipt unnecessary. Instead it links receipts together using a hash function. This prevents the adversary from retroactively changing an acquired receipt, which is necessary to change the corresponding vote, as this changes the hash value of this receipt. But this hash value is also contained in the subsequent receipt and an included when computing its hash value, so that checking a receipt also implicitly checks all receipts issued previously.

5.6.2 Hash Chains

It appears to be impractical to prevent the adversary from obtaining receipts. Therefore the goal must be to prevent the adversary to gain an advantage from stealing a receipt.

There are several methods to make a stolen receipt useless for the adversary. One approach is to ensure that the receipt stolen by the adversary is not the only copy, either by retaining one copy, as proposed by Ryan [Rya06, LR08], or by issuing additional copies to other voters. This is an inherent property of voting schemes based on the Farnel idea [ACvdG07]. Both methods require additional assumptions. Retaining an additional copy of each receipt only works if the adversary has no access to those copies. Issuing additional copies to other voters makes it harder for the adversary to collect all copies but implicitly assumes that the copies are unmodified.

Another approach is to immediately publish each receipt as soon as it is issued. This is done for example by Punchscan (cf. Section 2.5.5). This method requires a connection between the voting machine or scanner and a publishing medium, e. g. the bulletin board used by Punchscan. In most cases, this will be done via the Internet.

The hash chain method proposed here utilizes an idea that was proposed by Kiayias, Korman and Walluck in [KKW06] and also used in VoteBox [SDW08]. In both cases hash chains are used to document the integrity of internal states. The hash chain method for Bingo Voting uses a very similar technique not for internal but for public data, i. e. to ensure receipt integrity. This section describes the method for Bingo Voting, but it is adaptable to any cryptographic voting scheme that uses a computer to generate receipts.

The idea is to link a receipt to each receipt that was printed out before. For this a receipt is identified with a small value (a hash value for example) and printed onto the subsequent receipt. It is important that the value used for identification is depending deterministically on the content of the receipt, making the hash value of a publicly known cryptographic hash function a good choice.

When forming a receipt, the voting machine now hashes the complete content of the receipt and prints the hash value onto the receipt as well. To link each receipt with its predecessor the voting machine also prints the hash value of the previous receipt onto the current receipt and includes it into the information used to compute the hash value (see Figure 5.6). By integrating the hash value of the last receipt in the hashing of the new receipt the receipts form a chain. So with each receipt not only the corresponding vote but the integrity of all previous receipts can be verified. This prevents an adversary from manipulating a vote even if he gets hold of the corresponding receipt.

The first approach to link consecutive receipts is by printing the hash value of the last receipt on top of the current receipt (and include it in its hash value and thus forming a hash chain). When all receipts are published, it can easily be verified that for each receipt except the last there is a second one that carries its hash value on top and thus ensures the integrity of the complete hash chain.

There are two problems with this approach: The first is that the order of receipts becomes public, which we will discuss in Section 5.6.3. The second problem is that there still is a time window in which an adversary can get hold of a receipt, access the voting machine and change the vote, the corresponding receipt and its hash

value. If this happens before the next voter has cast their vote and received their receipt, this attack will not be noticed.

It is possible to modify the scheme so that the adversary is unable to modify any vote without risk. For this the hash value of the receipt that was just printed out must be fixed so that an adversary cannot modify it. One possibility is that the printer prints the hash value of a receipt onto the bottom, gives out the receipt and prints the same hash value on the top of the paper that will form the top of the next receipt. The voter can now compare both values and ensure that the next carries the correct link value. This assumes of course that the printer is not able to draw the paper back in and overwrite the link value or simply throw out the blank receipt with the hash value on top and start a new one after the voter has left the booth and before the next one enters. The second possibility is a display outside of the voting booth that displays the hash values of the last two receipts. So when a receipt is printed the display shows the hash values of this receipt and the one before. Both hash values are also printed on the receipt so the voter can easily check the correctness. Both hash values are displayed until the next receipt is printed, then the older hash value is replaced by the newer one, and the newer is replaced by the hash value of the receipt just printed out. This is the only time at which the display changes, any other change (which is necessary when an adversary wants to change a vote and the corresponding receipt) can be noticed by poll workers. This is of course a new assumption but as poll workers are necessary for all elections to control the correct proceeding of the election we think it is not unreasonable to assume that the display is watched with sufficiently high probability at a given time.

5.6.3 Hidden Hash Chains

One side effect of using a simple hash chain to link receipts is that the order in which votes are cast are published as they are required for the verification process. This gives additional information to an adversary and may be considered undesirable. However, as it is required that the receipts do not reveal the vote, publishing the receipts possibly even linked to the voters seems to be acceptable. An alternative is to use a commitment to the hash value of the previous receipt to link receipts instead of the hash itself. During verification instead of giving the order of the receipts the voting authority publishes a zero knowledge proof that for each commitment on a receipt there exists another receipt with this hash value. For this the voting authority publishes a proof of a shuffle with known content (e. g. using the proof technique proposed in Section 5.5.2) with the set of commitments printed onto the receipt as well as the list of the hash values of all receipts as input.

As the commitment is included in the hash it is impossible to commit to the hash value of the same receipt. Likewise it is impossible to break the chain or leave out a receipt as hash values will be verified during the tally phase.

This method masks the order in which receipts are issued and the only information that still becomes public knowledge is which receipts originate from the same voting machine.

The major drawback of this method is that it requires an additional proof, thereby increasing the size of the public data.

5.6.4 Relevance for Other Schemes

The problem of receipt stealing attacks is not limited to Bingo Voting. Most other cryptographic voting schemes are also vulnerable to this attack. A notable exception is Scantegrity II since the receipt of a Scantegrity II election contains only very little information so that a voter is able to remember this information and check the correct inclusion of their vote into the tally without the receipt. For most other voting schemes a vote corresponding to a stolen receipt will not be checked by the voter, allowing the adversary to alter this vote without risk of detection.

Fortunately the hash chain method described in Section 5.6 is not specific to Bingo Voting and only requires a computer to generate the receipt. It is easy to implement hash chains for all cryptographic voting schemes that use a voting machine to print the receipt.

6. Analysis and Evaluation

A thorough security analysis of a cryptographic voting scheme requires a security notion and a complete model. Unfortunately, a general model for the security of cryptographic voting schemes is still missing. There have been attempts for remote voting schemes [KR05, JCJ05, BHM08, Ond09, KRS10], but the results are not transferable to presence voting schemes.

There are multiple problems that make formalizing such a model hard. One such problem is finding the appropriate level of abstraction for real-world characteristics like physical assumptions or behaviour of human voters. Another problem is that it must be assumed that a human voter is unable to do anything but the most basic tasks during the voting process. A discussion of such a general security model for cryptographic voting schemes and its problems is out of scope of this work.

While the lack of such a model is unsatisfactory, it should be noted that a full formal analysis for the traditional paper election is also missing. Instead of a formal security model the security of a (cryptographic) voting scheme is typically evaluated considering security properties. Section 2.2 gives an overview over the security properties used for the evaluation of Bingo Voting in this chapter.

The security of all cryptographic voting schemes is based on a number of assumptions. The quality of a voting scheme is not only depending in the security it achieves but also on the nature and quantity of the assumptions required. Often assumptions are stated explicitly, but some assumptions are only made implicitly. This is dangerous, as implicit assumptions make an evaluation of the quality of a voting scheme hard. When an attack that breaks an implicit assumption is found, it may affect the security of the voting scheme without giving users a chance to prepare.

This chapter analyses the security properties of Bingo Voting and their relation with the required assumptions, and discusses and evaluates the effects of the improvements proposed in Chapter 5. For the evaluation of the improvements we will use three scenarios taken from real-world elections. These scenarios are described in Section 6.1. To evaluate the security properties of Bingo Voting, Section 6.2 presents the assumptions the correctness and verifiability of Bingo Voting are based on and assesses which are required to fulfil prevalent properties

of cryptographic voting schemes. Section 6.3 analyses the security properties of Bingo Voting and gives a detailed explanation which assumptions are required for each security property. Section 6.4 deals with practical considerations including an analysis of the usability of Bingo Voting and the size of the public data.

6.1 Scenarios for Evaluation

For the evaluation of the size reduction of the public data (cf. Section 5.5) and the length reduction of random numbers (cf. Section 5.2) we use three scenarios taken from three recent real-world elections:

(a) the German Bundestag election of 2009,

(b) the US presidential election of 2008, and

(c) the Indian general election of 2009.

For scenario (a) we find the following numbers in [Bun09]: $v = 62$ million voters, $n = 27$ candidates, and a turnout of $t = 0.7$. For scenario (b) we take the following numbers from [Uni08, Alb08]: number of eligible voters $v = 209$ million, number of candidates $n = 36$ and turnout $t = 0.63$. For scenario (c) we have $v = 714$ million voters, $n = 364$ candidates and a turnout of $t = 0.58$, taken from [Sid10] and [Tha09].

We use the scenarios to estimate the probability of a collision for different lengths of random numbers and to assess the size of the public data required for verification. The Indian general election in 2009 (scenario (c)) was conducted using 1 368 430 voting machines for 714 million eligible voters [Sid10, Tha09] which means circa 520 voters per voting machine. Using this number we estimate the number of voting machines that would have been used for the German Bundestag elections in 2009 (scenario (a)) with 124 000, and for the United States presidential elections in 2008 (scenario (b)) with 418 000.

6.2 Assumptions Required for Bingo Voting

The security properties of all cryptographic voting schemes are based on a number of assumption, some of which are complexity theoretical assumptions typical for cryptographic protocols, but many are assumptions that make theoretical considerations applicable to real-world elections.

It is important to be aware of all assumptions that are required for certain security properties of a cryptographic voting scheme. To assess the security of a cryptographic voting scheme it is also necessary to assess the required assumption. An implicit assumption may be the target of an attack that breaks the security of a cryptographic voting scheme without notice.

A good example for this is the receipt stealing attack described in Section 5.6.1. Many cryptographic voting schemes implicitly make the assumption that the adversary does not know which votes are checked. This implies that the adversary takes a certain risk for each vote that is manipulated. This makes changing a significant number of votes apparent with high probability. The receipt stealing attack, for which the adversary obtains receipts and is now able to predict with high confidence that the corresponding votes will not be checked, invalidates this

assumption. This makes additional security measures necessary to protect against such an attack.

This section gives an extensive list of assumptions required for the security of a Bingo Voting election. Section 6.2.1 briefly discusses assumptions that are required for any election and are independent of the voting scheme used. The voting booth assumption discussed in Section 6.2.2 is also essential for voter privacy in any election, but since Bingo Voting introduces a voting machine it is discussed separately. Section 6.2.3 discusses the assumption that voters participate in the verification process. Section 6.2.4 presents details on the central assumption required for the correctness of Bingo Voting, namely that the trusted random number generator is uncorrupted. Section 6.2.5 presents the assumption that an authentic receipt is recognizably different from a forged receipt. Section 6.2.6 describes the discrete logarithm assumption, the only complexity-theoretical assumption required for Bingo Voting.

6.2.1 General Assumptions

A cryptographic voting scheme typically does not specify all technical details and tasks of a real-world election. Tasks like voter registration and authentication, checking voter eligibility, managing candidate lists, and organizing and arranging polling stations and poll workers are essential for an election but not part of the specifications of a cryptographic voting scheme.

This section gives a brief overview over the technical and organizational tasks that are required for an election in addition to the specifications given by Bingo Voting.

Probably the most important assumption for the correctness of a Bingo Voting election is that only eligible voters get access to a voting machine, and that each voter only casts a single ballot. At the same time, we have to assume that no eligible voter is prevented from participating in the election. This includes making the date of the election public and providing a sufficient number of polling stations for all voters. Similarly we have to assume that the candidate list is complete and public.

The assumption that every eligible voter is able to participate in the election also implies that each voter is able to use the interface of the voting machine employed by Bingo Voting. One big advantage of Bingo Voting is that the voting scheme is independent of the user interface used (see Section 6.4.1). This allows the voting authority to choose an appropriate user interface.

Most of these concerns are independent of the voting scheme used and have to be solved for any election. This means that these problems have already been solved for traditional elections and an election employing Bingo Voting is able to use the same solutions without any major modification. An important part of this is the deployment of poll workers or auditors that represent the public (or at least competing candidates) and supervise the correct procedure of the election. Another method to ensure transparency is making the list of eligible voters public and also publish the polling station records that state if an eligible voter voted. While this helps verifying the assumption that only eligible voters voted, it still requires the correctness of the polling station. It also decreases voter privacy since it makes the fact whether a voter attended public.

For an election employing Bingo Voting we assume that the list of candidates

is complete and publicly known, that each voter may choose freely to participate or not, and that only eligible voters are able to cast at most one vote.

6.2.2 Voting Booth Assumption

In a secret election, voters fill their ballots in a designated area designed to protect privacy during the voting process. For voter privacy it is essential that all voters are unobserved during the voting process when they mark their ballot or enter their choice into a voting machine. This assumption is important for all secret elections, but it becomes stronger if the voting scheme includes a voting machine which may leak information about the voter's behaviour.

For Bingo Voting the assumption that the inside of the voting booth is unobservable by the adversary is as essential as for other voting schemes. The assumption that the voting machine used to record the voter's choice and generate the receipt does not transmit any information of the vote outside of the voting booth extends this assumption. A voting machine may even leak such information without being manipulated, simply due to side channels like electromagnetic radiation of the screen. These side channels remain an open problem for all voting machines.

For Bingo Voting we assume that the voting booth assumption holds. This means that the adversary is unable to observe a voter, the voting machine or the trusted random number generator inside the voting booth during the voting process, and that neither the voting machine nor the trusted random number generator give any information (intentionally or unintentionally) to the adversary. This does not include the information printed on the receipt that is handed to the voter or information learned from the tally.

6.2.3 Verification Assumption

The goal of most cryptographic voting schemes is that each voter is convinced that either the tally is correct or some verification step fails. But the goal reached by many schemes is that each interested voter is able to check whether their vote was included correctly in the tally. These two slightly different notions are only equivalent if each voter checks the correctness of the tally.

For a practical application this assumption is not realistic. Many voters are probably not able to verify the tally, especially for cryptographic voting schemes whose verification processes often include cryptographic proofs, or may even be not interested in doing so. This typically leads to the assumption that while not each vote is checked, still only very few votes may be changed by the adversary without being noticed. It is also assumed that a small number of altered votes does not change the tally sufficiently to change the result of the election.

This assumption is based on two arguments. The first is that in traditional paper based elections small counting errors are frequent and that to change the result of an election typically a large change in the tally is required. The second argument is that while not every vote is checked the adversary does not know which voter will check their vote and that a sufficiently large number of voters will do this. This means that for each vote the adversary changes there is a certain risk of detection that is large enough that a significant change of the tally is probably detected.

There are several problems with this argumentation. The biggest problem is that many of these factors are unknown and probably very specific to each election.

This is especially true for the number of votes that may be changed without changing the result of the election. In a small election only one or two votes may change the outcome, but also for large elections the result may hinge on less than a thousand votes. Since the chance of detection only depends on the number of changed votes and is independent from the total number of votes small elections are more at risk. The probability that a small change in the tally changes the result of an election is much higher for small elections than for large elections. The fraction of voters that will check their receipts is also very hard to estimate.

Bingo Voting shares this problem with many cryptographic voting schemes. The security of the final result depends on the assumption that an adversary changing votes is caught with high probability. This in turn requires that all voters check their receipts or alternatively that the adversary is unable to predict with a high certainty which receipt will be checked. If this is the case, a verifier is convinced that the set of receipts that is published is identical to the set of receipts issued during the voting phase or any "large" difference is noticed and challenged. In many cases, this assumption is made implicitly, and only few cryptographic voting schemes are actually protected against attacks targeting this assumption. Bohli et al. analyse the threat of this attack in [BHK+09] and find that only Scantegrity II is inherently protected as the information that the receipt contains is so short that a voter may remember it even when losing the receipt itself.

If the adversary does not know if a certain receipt is checked, he risks detection with every vote changed. In most elections it is necessary to change a number of votes to change the result (cf. Section 2.2). While the ultimate goal of each cryptographic voting scheme is to guarantee a perfectly correct tally, this goal is unrealistic. The fact that not each and every voter is guaranteed to check their receipt gives the adversary a chance to change a single vote without being detected.

The verification assumption states that for each vote manipulated the adversary risks detection. This probability is high enough that the probability that the adversary is able to change a sufficient number of votes to change the result of the election is sufficiently small. This includes the assumption that small changes in the tally do not change the result of the election and that cryptographic proofs are checked by a capable public.

6.2.4 Random Number Generator

The original work on Bingo Voting [BMQR07a] proposes using randomness from a mechanical source, for example a bingo cage, to generate the random numbers necessary for the voting scheme. While this idea is alluring, there are several problems with this. The first is that the random number generated must be readable by the voting machine to generate the receipt. This is a technical problem which appears to be solvable, for example by using balls with bar codes. The second problem is that the randomness must remain random even when the voter misbehaves, i. e. the voter must not be able to influence the result of the process. This is necessary to ensure the coercion resistance as the voter must not be able to generate a recognizable "random" number. This, too, is a technical problem that appears to be solvable.

The last problem with a mechanical random number generator is the amount of entropy that is required for the random numbers. Drawing a single ball from

a typical bingo cage with 75 balls generates $\log_2 75 \approx 6.23$ bits of entropy. Section 6.3.5 argues that a real-world election requires up to 45 bits of entropy for each random number which is hard to obtain by drawing a single ball from a bingo cage (this would require a bingo cage with $2^{45} \approx 35 \cdot 10^{12}$ different balls). The alternative is drawing more than one ball from the cage, either with or without putting back the ball after drawing. As the resulting random number should be visible to the voter for the verification process (cf. Section 3.4), putting back the balls after each drawing is probably not practical.

While drawing more than one ball greatly increases the entropy of the resulting random number, the drawing process becomes more complex and time consuming. To achieve more than 30 bits of entropy the random number generation process would require drawing five balls (without putting them back) from a cage of 100 balls, each labelled with two digits from 00 to 99. This gives circa 33 bits of entropy and would be sufficient for most real-world elections (cf. Section 6.3.5.2). Please note that these numbers are valid only for the scheme using the improvements described in Section 5.2. The entropy requirement becomes more demanding for more complex elections as discussed in Section 6.3.5. Finding a mechanical device that reliably and comprehensibly produces randomness of sufficiently high entropy is currently an open problem. These requirements become more demanding for complex elections in which the voter is able to distribute more than one vote. This may be solved with one trusted random number generator for each vote the voter distributes, however.

Electrical random number generators provide an alternative to mechanical random number generators. They often use the randomness generated by Johnson-Nyquist noise [Joh28, Nyq28] and provide several advantages over physical random number generators. They offer a higher rate of entropy, often provide their data in digital form and established implementations for smart cards (e. g. signature cards) exist. Their main disadvantage is that the generation of randomness is not visible to the human eye, making it hard to verify the true randomness of their results.

The trusted random number generator assumption is central to Bingo Voting and actually consists of two assumptions. The first is required for correctness and states that before the publication of the dummy random numbers the voting authority is unable to predict the random numbers drawn by the trusted random number generator during the voting phase better than by inference from the publicly known distribution. The assumption that the trusted random number generator is uncorrupted and actually draws and displays a fresh random number for each voter remains the one most important assumption for the correctness of Bingo Voting. The second assumption is required for receipt-freeness and states that the distributions of dummy random numbers and random numbers drawn during the voting phase are indistinguishable.

6.2.5 Receipt Authenticity Assumption

When a dispute about a single vote arises after the voting phase, i. e. when a voter notices that their receipt is not included correctly in the list of receipts, the voter is asked to produce the receipt issued by the voting machine. If the voter is able to produce an authentic receipt that was not included in the list, this indicates that the corresponding vote was manipulated.

The dispute resolution procedure assumes that the voting authority is able to verifiably discern between an authentic receipt and a forgery. Measures that support this assumption are discussed in Section 5.4.2.

6.2.6 Discrete Logarithm Assumptions

Bingo Voting requires a complexity theoretic assumption for the binding property of the commitments used for the dummy votes and the proofs of shuffles of known content.

The UHDLCs (see Section 2.4.2.1) use a group in which the discrete logarithm is hard to calculate. The commitment scheme requires two generators and is binding under the assumption that the sender is unable to compute the discrete logarithm of one generator in respect to the other. This not only requires that the discrete logarithm is hard to compute but also that the generators are chosen at random. There are two approaches to choose two generators at random. One approach is using a secure multiparty computation with at least one uncorrupted party to choose the generators using the combined randomness, the other is using a random beacon [Rab83, CH10] to retrieve public randomness for choosing the generators.

The discrete logarithm assumption states that the two generators chosen for the UHDLC scheme are chosen so that the discrete logarithm is unknown, and it is infeasible to compute it.

6.2.7 Hash Function Assumptions

The hash chain method proposed in Section 5.6 employs a cryptographic hash function to generate a short bit string representing the content of a receipt. By incorporating the hash value of the last receipt two consecutive receipts are linked and subsequently all receipts form a hash chain.

In order for this to protect the vote corresponding to the receipt from being changed, the hash function used must have certain properties. The collision resistance property guarantees that altering the content of a receipt leads to a different hash value for the receipt. It prevents the adversary from creating two receipts with the same hash value. One would be the receipt being printed that correctly represents the voter's choice, the other would be the manipulated receipt published in case the adversary obtained the printed receipt. If the dummy votes are prearranged as suggested in Section 5.2, the second-preimage resistance property is probably sufficient to ensure this since the correct receipt which is issued to the voter is determined by the cluster that is to be used and the voter's choice.

6.3 Security Analysis of Bingo Voting

Many cryptographic voting schemes promise verifiable correctness as well as voter privacy, even when an adversary is present. A general security model is currently missing, most evaluations use security properties like end-to-end security, receipt-freeness, and coercion resistance. This section analyses the security of Bingo Voting based on the assumptions presented in Section 6.2.

Section 6.3.1 discusses how Bingo Voting protects voter privacy and how it prevents vote buying and coercion. Section 6.3.2 evaluates the correctness and end-to-end verifiability of Bingo Voting. Section 6.3.3 discusses the properties of Bingo Voting in case of a dispute. Section 6.3.5 looks at the consequences of a random number being drawn more than once for the security of Bingo Voting.

6.3.1 Voter Privacy

For voter privacy we assume that the voting booth assumption (see Section 6.2.2) holds, i. e. that the adversary is unable to observe the voter during the voting process and that the voting machine does not communicate with the adversary. In the original Bingo Voting scheme the voting machine was able to use the choice of dummy random numbers for the receipt to leak information about the voter's choice. If the fresh random number for example was even and the voting machine chose only odd dummy random numbers for the receipt, the voter's choice would become apparent immediately. The method of prearranging dummy votes actually provides a countermeasure against this since the voting machine is not longer able to choose the dummy votes that are used to generate the receipt based on the voter's choice but is forced to use a set of dummy votes specified in advance.

The voting authority plays a prominent and important role in an election. In most cryptographic voting schemes it possesses sufficient information to break voter privacy and learn each voter's choice. This is also true for Bingo Voting. In the original paper [BMQR07a] the authors suggest to solely rely on the voting machine for the preparation phase (cf. Section 3.3.1). This means that the voting machine possesses all information that may break voter privacy. Implicitly, using this approach assumes that the voting authority does not access this information. This is probably not practical for large elections since the preparation of a Bingo Voting election requires the generation of a large amount of randomness (to generate dummy votes) and access to the Internet for publishing dummy votes.

The most important assumption for providing voter privacy is the voting booth assumption (cf. Section 1). In short it states that the voter is unobserved inside the voting booth. This assumption is necessary for all cryptographic voting schemes as well as for a traditional election with paper ballots. But by using voting machines an additional element is introduced that threatens the voting booth assumption.

The voting booth assumption for voting schemes that use voting machines states that not only is the voter unobserved inside the voting booth during the voting process. In addition it requires that the voting machine does not transmit the voter's choice outside the voting booth or stores it together with information that allows the adversary to connect the choice to the voter (for which it would be sufficient to store the time or the order in which the votes are entered).

One big problem for voting machines in general are side channels, most prominently electromagnetic radiation [FDBV11]. It was shown that this side channel is sufficient to gain information about the voter's choice even from voting machines employed in real-world elections (in this case the national elections of the Netherlands) which were not designed to specifically transmit the voter's choice to break voter privacy. Side channels are, however, a technical problem and out of scope of this work.

From the point of view of vote secrecy or coercion resistance, a voting authority that possesses all secrets to revoke the vote secrecy is unfavourable. The solution of distributing the voting authority onto several parties by means of secure multiparty computation might not completely solve the problem, because (at least for Bingo Voting) all data must be available to the voting machine, so a corrupted voting machine still poses a threat to secrecy.

One option to avoid an omniscient voting authority is to have the election prepared inside the voting machine, as mentioned in the description of Bingo

Voting in Section 3.3.1. This means that each voting machine is treated as a separate election and produces its own result (cf. Section 5.1.4). This reduces the amount of vote secrecy that is achievable at all. The smaller the sample of voters the higher is the probability that information about the vote cast by a single voter can be deduced from the result (for example if a candidate receives no votes). This also trades the assumption that the voting authority is uncorrupted for the assumption that the voting authority has not illegitimate access to the voting machines. The appraisal which assumption is preferable depends on the specifics of a real-world election and is out of scope of this work.

6.3.1.1 Receipt Freeness

In an election employing Bingo Voting each voter receives a receipt during the voting phase (see Section 3.3.2). In order to be receipt-free, the receipt issued by Bingo Voting must not leak any information about the voter's choice. This requirement is necessary in addition to voter privacy. For this reason this section concentrates on information contained on the receipt.

For Bingo Voting the receipt consists of the list of candidates together with the random numbers representing the voter's choice (see Figure 3.1). The random number that represents the voter's choice was generated by the trusted random number generator during the voting process, the other random numbers are from dummy votes for the corresponding candidates. It is easy to see that if the adversary is able to discern between fresh random numbers and dummy random numbers the receipt leaks information about the voter's choice. So for receipt freeness the distribution of fresh random numbers and dummy random numbers has to be indistinguishable. For this it is particularly essential that the dummy random numbers remain secret. This means that if any party that knows the dummy random numbers, i. e. voting authority and voting machine, is corrupted (and able to communicate with the adversary), receipt-freeness is broken.

If the dummy votes that are used to generate the receipt are not prearranged (see Section 5.2), a corrupted voting machine is able to choose dummy random numbers in a recognizable way. If the dummy votes are prearranged, a corrupted voting machine is no longer able to use this side channel without being detected.

This means that Bingo Voting is receipt-free if the adversary does not learn which random numbers on a receipt are dummy random numbers. For this the following requirements have to be fulfilled:

- The voting authority is uncorrupted.
- The distribution of the dummy random numbers is indistinguishable from the distribution of the random numbers generated during the voting phase.
- The voting machine is either uncorrupted or unable to communicate with the adversary directly and the receipt contains no side channels (because the generation is deterministic using prearranged dummy votes).

6.3.1.2 Coercion Resistance

For coercion resistance Bingo Voting has to provide in addition to receipt freeness that a voter may not gain an advantage in proving their vote to the adversary by deviating from the voting protocol. For Bingo Voting the analysis is rather short, since the voter is not confronted with the voting scheme before the vote

was cast. There is no way for the voter to influence the form of the receipt in a specific way. This means that the receipt-freeness of Bingo Voting directly implies coercion resistance.

6.3.2 Correctness and Verifiability

The correctness of Bingo Voting is based on the mechanism a Bingo Voting election uses to keep track of the dummy votes. During each voting process, whenever a voter casts a vote, all but one candidates lose one dummy vote. That means that after the vote was cast, one candidate possesses one dummy vote more, relative to all other candidates, than before. For the remainder of this section we assume that each eligible voter has cast a vote in the election.

The individual verification of each voter's receipt described in Section 6.3.2.1 ensures that the candidate the voter voted for does not lose a dummy vote during the voting process. The global verification described in Section 6.3.2.2 ensures that all candidates have received the same number of dummy votes and that for each receipt all but one candidate lose one dummy vote. The implication is that the number of remaining dummy votes for each candidate is equal to the number of votes received if the number of dummy votes at the beginning of the voting phase was equal to the number of voters.

6.3.2.1 Individual Verification

Inside the voting booth, the voter is able to verify that the random number associated with the candidate of their choice is not a dummy random number by checking the display of the trusted random number generator. This is only true, however, under the assumption that the trusted random number generator is uncorrupted and displaying a fresh random number and that no collision occurred. Section 6.3.5 argues that the length of the random numbers have to be chosen so that collisions are sufficiently improbable.

The second part of the individual verification is the comparison of the copy of the receipt published by the voting authority with the original receipt. By checking that their receipt was published without changes each voter verifies that their vote was included in the tally and concludes the individual verification process.

The individual verification process assures the voter that during a voting process the elected candidate does not lose a dummy vote. The required assumptions for this are the discrete logarithm assumption, which is needed for the binding property of the commitments forming the dummy votes and for the proofs, and the trusted random number generator assumption.

6.3.2.2 Global Verification

The global verification consists of a number of proofs that show the correctness of the tally. All of these proofs are essentially proofs of shuffles of known content (see Section 1).

The first proof shows that all candidates received the same number of dummy votes during the preparation phase. In the original Bingo Voting scheme the voting authority proves this during the preparation phase when the dummy votes are published. Section 5.5.3 proposes proving this after the voting phase. In both cases the correctness of the proof is based on the binding property of the commitment and therefore by the assumption that the discrete logarithm in the group G used for the UHDLCs (cf. Section 3.1.2 and Section 6.2.6) is hard.

The second proof shows that all but one random number on each receipt is a dummy random number and that all dummy votes used are for different candidates and that none is for the elected candidate. This means that all but one candidate lost a dummy vote. The individual verification ensures that the elected candidate did not lose a dummy vote, so all other candidates must have lost one. Since the second proof is essentially the same as the first one, its security is also based on the discrete logarithm assumption.

In an election that uses prearranged dummy votes as described in Section 5.2 the unused dummy votes are not simply opened but instead shuffled and then opened. This includes a proof of a shuffle of known content to verify that the unused dummy votes were used correctly. The correctness of this proof is also based on the discrete logarithm assumption.

When the election uses hash chains as described in Section 5.6 the verification that the hash chain was not broken is also part of the global verification. The security of the hash chain is based on the collision resistance of the hash function that is used.

6.3.3 Dispute Freeness

For Bingo Voting a dispute may arise at two points during the verification process. The voter may notice that the receipt is incorrect inside the voting booth or that the receipt was not published correctly after the voting phase. If any part of the proof fails, this does not lead to a dispute since the data required for the verification is public and may be checked by any auditor.

For Bingo Voting, the verification inside the voting booth is critical. The protocol presented in Section 5.4.1 offers a possibility for the voter to prove a manipulation without revealing their choice. This protocol is very sensitive, however, and may be too complicated for use in real-world elections.

If the dispute rises after the voting phase, the dispute freeness of Bingo Voting is mostly based on the assumption that the question whether a receipt is authentic or not can be resolved reliably (see Section 6.2.5). If this assumption is met, Bingo Voting allows the voter to prove a manipulation without disclosing any information about their choice.

6.3.4 Robustness and Recovery

The Bingo Voting scheme allows voters and auditors to notice manipulations but offers no recovery mechanism without violating voter privacy. This means that a single manipulation may force the voting authority to annul the complete election.

6.3.5 Collisions

The random numbers used by Bingo Voting are assumed to be long enough to avoid drawing the same random more than once during an election. We call the occurrence of two or more identical random numbers a *collision*. For the correctness it is necessary to choose the length of the random numbers sufficiently long in order to make collisions unlikely.To understand why it is necessary to choose the length of the random numbers of Bingo Voting accordingly and why we want to avoid collisions we look at what happens when a collision occurs.

There are three types of collisions, collisions between dummy random numbers that already occur during the preparation phase, collisions between fresh random

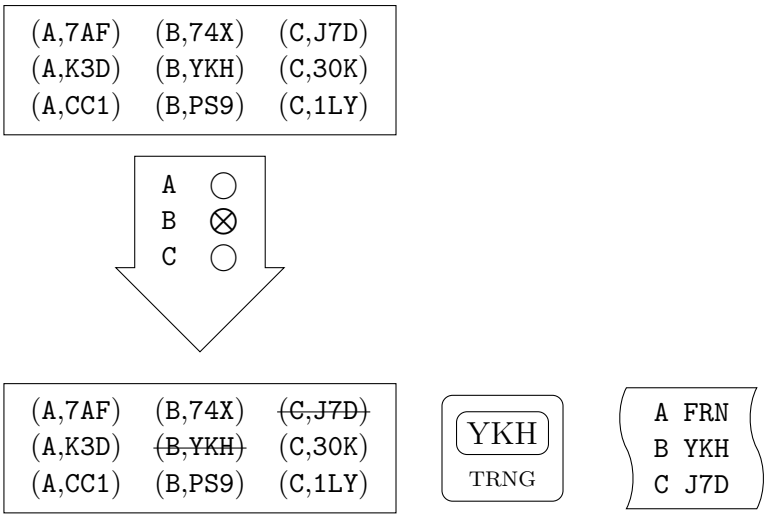


Figure 6.1: A collision occurring during the voting process allows a corrupted voting machine to undetectably change a single vote. The figure depicts the set of dummy votes before (top) and after (bottom left) a voter chose candidate B. The collision between the fresh random number displayed by the trusted random number generator (TRNG, bottom middle) and a dummy random number for candidate B allows the corrupted voting machine to count the vote cast for candidate A (A lost no dummy vote). The TRNG displays the random number printed next to the candidate B on the receipt (bottom right), so the voter is (falsely) convinced that the vote was counted for candidate B.

numbers generated by the trusted random number generator during the voting process, and collisions between a dummy random number and a random number. Most of these collisions have not consequences for the correctness, but when a fresh random number is identical to an unused dummy random number for the elected candidate, a corrupted voting machine is able to change the corresponding vote without the voter noticing.

In this section we have a close look collisions and discuss the consequences for the security of an election using Bingo Voting. It is important to stress that a collision only has consequences for the correctness of an election if the voting machine is corrupted. For the remainder of this section we assume that this is the case and show how a voting machine is able to change a single vote when a collision occurs. We also determine the probability with which such a collision occurs and compare the probabilities for the original version of Bingo Voting and of the scheme using prearranged dummy votes as described in Section 5.2. This section also presents the probabilities of collisions occurring for several lengths of random numbers for three real-world scenarios.

6.3.5.1 Consequences of Collisions

The original description of Bingo Voting [BMQR07a] assumes that the random numbers used are long enough and collisions are negligible. If a collision occurs during the voting process, however, a corrupted voting machine potentially changes one vote.

There are two types of collisions. A collision that happens during the prepa-

	(a)		(b)		(c)	
	$\mathbb{P}^{\text{old}}_{\text{Elec}}$	$\mathbb{E}^{\text{old}}_{\text{Elec}}$	$\mathbb{P}^{\text{old}}_{\text{Elec}}$	$\mathbb{E}^{\text{old}}_{\text{Elec}}$	$\mathbb{P}^{\text{old}}_{\text{Elec}}$	$\mathbb{E}^{\text{old}}_{\text{Elec}}$
$l = 15$	1	431 113	1	1 378 259	1	4 694 498
$l = 30$	1	13	1	42	1	143
$l = 45$	$4.014 \cdot 10^{-4}$	$4.015 \cdot 10^{-4}$	$1.283 \cdot 10^{-3}$	$1.284 \cdot 10^{-3}$	$4.363 \cdot 10^{-3}$	$4.372 \cdot 10^{-3}$

6.2.1: original Bingo Voting scheme

	(a)		(b)		(c)	
	$\mathbb{P}^{\text{new}}_{\text{Elec}}$	$\mathbb{E}^{\text{new}}_{\text{Elec}}$	$\mathbb{P}^{\text{new}}_{\text{Elec}}$	$\mathbb{E}^{\text{new}}_{\text{Elec}}$	$\mathbb{P}^{\text{new}}_{\text{Elec}}$	$\mathbb{E}^{\text{new}}_{\text{Elec}}$
$l = 15$	1	1324	1	4018	1	12 638
$l = 30$	$3.961 \cdot 10^{-2}$	$4.042 \cdot 10^{-2}$	0.115	0.113	0.320	0.386
$l = 45$	$1.234 \cdot 10^{-6}$	$1.234 \cdot 10^{-6}$	$3.742 \cdot 10^{-6}$	$3.742 \cdot 10^{-6}$	$1.177 \cdot 10^{-5}$	$1.177 \cdot 10^{-5}$

6.2.2: improved Bingo Voting scheme

Figure 6.2: Probabilities of at least one collision occurring during an election for different random number lengths k and different election scenarios: (a) German Federal elections, 2009, (b) USA presidential election, 2008, and (c) Indian general election, 2009.

ration phase has no consequences. A collision that happens during the voting process only affects the election if the fresh random number is identical to an unused dummy random number of a dummy vote for the elected candidate. But if this is the case, the voting machine may count this vote for any other candidate.

Figure 6.1 visualizes the consequences of a collision occurring during the voting process. If the trusted random number generator draws a fresh random number matching the dummy random number of an unused dummy vote for the elected candidate, the voting machine is able to use this dummy vote for receipt and to count this vote for any other candidate without the voter noticing the manipulation. The voting machine generates the receipt using the dummy vote for the elected candidate and a new random number, drawn by the voting machine, for a candidate of the voting machine's choice. The voter will be convinced that the vote was counted correctly as the trusted random number generator displays the random number written next to the candidate elected by the voter.

This problem is impossible to detect or prevent. It is thus essential to make sure that such collisions occur only with very small probability. To alleviate this problem we propose to prearrange dummy votes as described in Section 5.2.

6.3.5.2 Probability of Collisions

In this section we discuss the length of the random numbers that is required to avoid collisions with a certain probability. To compare our changes to the original Bingo Voting protocol and to assess the length requirements, we calculate the probability of collisions for elections employing the original and the improved scheme for the scenarios presented in Section 6.1. We state the probabilities relative to random numbers with a length of l bits.

For the original scheme, the probability of a collision occurring during a single voting process of a voter is

$$\mathbb{P}^{\text{old}}_{\text{V}} = \frac{\hat{d}}{2^l} \tag{6.1}$$

where $\hat{d}$ is the number of remaining dummy votes for the candidate of the voter's

choice. Since the number of remaining dummy votes for any candidate changes during the voting phase (whenever a candidate is chose all other candidates lose a dummy vote) we give a lower bound for the probability for a collision occurring during an election on a single voting machine. This number depends on the elected candidate and the number of voters that have cast their votes for the same candidate before. This makes it impossible to give an exact probability of an collision occurring for the original version of Bingo Voting. In order to have an estimation for the probability of a collision for the original Bingo Voting scheme, we estimate the probability as follows.

When the ith voter votes, there are at least $v-(i-1)$ dummy votes left for any candidate left if there were v dummy votes for each candidate at the beginning. Therefore,

$$\mathbb{P}_{\mathrm{VM}}^{\mathrm{old}} = \prod_{i=1}^{v\cdot t} 1 - \frac{(v_{\max} - (i-1))}{2^l} \tag{6.2}$$

is an upper bound for the probability that no collision occurred on a voting machine that was prepared for v voters after $v \cdot t$ voters voted ($0 \leq t \leq 1$ being the turnout of the election). The expected value of the number of collisions occurring on a voting machine is

$$\mathbb{E}_{\mathrm{VM}}^{\mathrm{old}} = \sum_{i=1}^{v\cdot t} \frac{v-(i-1)}{2^l} \tag{6.3}$$

using the same simplifications. Again, this is an upper bound for the expected value. For an election with m voting machines the probability of a collision occurring is

$$\mathbb{P}_{\mathrm{Elec}}^{\mathrm{old}} = 1 - \left(\mathbb{P}_{\mathrm{VM}}^{\mathrm{old}}\right)^m \tag{6.4}$$

and the expected number of collisions is

$$\mathbb{E}_{\mathrm{Elec}}^{\mathrm{old}} = m \cdot \mathbb{E}_{\mathrm{VM}}^{\mathrm{old}} \tag{6.5}$$

during a complete election. Both the probability and value for the expected number of collisions are probably lower bounds, but the exact values are hard to determine since they depend on the number of voters per voting machine, the voters' choices (and the order in which the votes were cast) and the distribution of voters to voting machines. For our analysis we used the estimates presented here and assumed that the voters were evenly distributed among the voting machines.

For an election using our improved scheme we calculate the probability of a collision as

$$\mathbb{P}_{\mathrm{Elec}}^{\mathrm{new}} = \left(\frac{1}{2^l}\right)^{v\cdot t} \tag{6.6}$$

for $v \cdot t$ votes cast. Note that the probability is independent of the number of voters the election was prepared for and of the number of voting machines or distribution of voters to voting machines. The reason for this is that each cluster of dummy votes may be treated as a single election since a collision is only relevant if it occurs between the fresh random number and the dummy random number of the candidate the voter chose and that is part of the cluster the voting machine has to use for this voter's receipt. Consequently, the expected number of collisions for the improved scheme is

$$\mathbb{E}_{\mathrm{imp}} = v \cdot \left(\frac{1}{2^l}\right). \tag{6.7}$$

Figure 6.2 lists the probabilities and expected number of collisions for the three scenarios taken from real-world elections (see Section 6.1). It shows that our suggestions greatly decrease the probability of a collision occurring. It also shows that a length of 30 bits for the random numbers is sufficient for very large elections with hundreds of millions of voters since the expected number of collisions is less than one for the complete election.

6.4 Practical Considerations

In addition to the security properties of Bingo Voting, several additional properties are important to consider to assess the applicability of Bingo Voting for real-world elections. Chapter 5 has presented several methods to improve the practicability of Bingo Voting by increasing usability and reducing the size of public data.

This section discusses these properties which are not directly security related but are essential for a large-scale deployment of Bingo Voting. Section 6.4.1 discusses the usability of Bingo Voting and compares it to the usability of several other cryptographic voting schemes. Section 6.4.2 analyses the size of the public data required for the verification of a Bingo Voting election and evaluates the improvements proposed in Section 5.5. A very important aspect of applicability are the legal requirements for cryptographic voting schemes. Section 6.4.3 presents and discusses the ruling of the German Federal Constitutional Court on the use of voting machines for the German Federal Election in 2005. This ruling makes a clear statement on the legal requirements for election schemes to be allowed to be employed in political elections in Germany.

6.4.1 A Comparison of Usability

The usability of a cryptographic voting scheme is important for the acceptance. The equality of all votes also demands that no voter may be influenced (by being confused by the voting scheme) or even deterred from voting. This implies a high hurdle for the usability of which the details are unfortunately unclear. Defining the exact requirements for usability of a cryptographic voting scheme is out of scope of this work.

This section will instead present a comparison of three verifiable voting schemes, Prêt à Voter, Punchscan and the scheme by Moran and Naor, to Bingo Voting, focusing on usability under realistic conditions. One aspect is the effort needed for the actual voting process, the other is the additional cost for the voter to ensure correctness.

One important question is at which point of the voting process the voter first gets into contact with the cryptographic mechanisms, i. e. when does the voter first experience a difference to the "normal" voting. There are studies that hint that voters may be influenced by the voting procedure, and that this influence is bigger if the voting scheme is more complicated [BLS+03, THP+05, HNH+07]. The results presented in this section are published in [BHMQ+08]

6.4.1.1 Prêt à Voter

As Prêt à Voter, like Punchscan, is a paper based system, the voter will not vote at a computer but cast a paper ballot. But in both cases this paper ballot looks different than the traditional paper ballot (see Sections 2.5.4). Both schemes use

a permutation to encrypt the votes on the receipt and both must mark the ballot in a way that the permutation can be inverted and the vote can be reconstructed.

Prêt à Voter uses a special paper ballot on which the candidates are printed in a random order. In contrast, normal paper ballots have a fixed order of candidates. This is a change to the normal paper ballot for which the order in which the candidates appear is fixed. This may also pose a conflict with election laws that regulate the order of appearance of the candidates on the paper ballot.

For Prêt à Voter the voter has to verify that this ballot is authentic and "correct", i. e. the encrypted permutation printed on the ballot corresponds to the permutation used for the candidates so that the vote can be correctly reconstructed. This is normally done by presenting two ballots to the voter, who chooses one that is to be verified and uses the other for voting. It is possible to do this by handing out two paper ballots to the voter (to prevent attacks the ID and permutation are hidden either by a scratch field or simply by showing the ballots upside-down), the voter chooses one and for this the correctness of the ballot is proven by opening the permutation.

This cut-and-choose approach catches a manipulation of n ballots with probability $1 - 2^{-n}$, making an undetected change of a large number of votes unlikely. There are two problems with this approach: First the number of paper ballots needed is doubled, and second the proof is either time-consuming or requires trust into an independent computer which may or may not be corrupted.

Filling out the Prêt à Voter ballot may be more time consuming than it is for a normal paper ballot, as the order of appearance is random so the voter has to spend some time to find their candidate. This is especially true for ballots containing many candidates, for elections in which the voter distributes several votes to more than one candidate and for elections with a ranking of candidates.

When the voter has completed the voting process they go to a poll worker and hand over the ballot upside-down without showing their vote. The poll worker removes the part with the names of the candidates and destroys it, e. g. using a shredder. The remaining part is scanned (for electronic counting) and signed with a digital signature.

If the ballot ID is protected by a scratch field, it must be removed before scanning, but after shredding the part with the names of the candidates. This poses some practical problems as the poll worker assisting in shredding and scanning must verify that the scratch field is intact without seeing the whole ballot as this would reveal the vote.

6.4.1.2 Punchscan

Punchscan also uses a paper ballot consisting of two parts. Here the voter has to find their candidate (order of appearance may be random or fixed), read a character printed next to it and find the same character in the marking area. The top layer has holes through which the second layer, lying underneath, displays the same characters that appear next to the candidates in a random order (one character per hole). The voter now marks the hole which shows the character corresponding to the candidate of their choice. This results in a clear mark on both the upper and lower layer of the ballot. The procedure of finding the candidate, reading the character and then again finding the character may be challenging and time-consuming for some voters. The time needed increases significantly if

there is a large number of candidates or if more than one vote can be distributed. Also the error rate of the voting procedure is probably affected.

After the voter has marked their ballot, they will go to a shredder, destroy one of the two layers and scan the other. If the ID of the ballot is protected by a scratch field for security reasons then this results in similar problems as Prêt à Voter.

6.4.1.3 Voting Scheme by Moran and Naor

Both Bingo Voting and the scheme by Moran and Naor use a computer to cast the vote. So naturally both schemes show significant difference to a traditional paper and ballot voting.

However, compared to other voting machines the differences only become apparent to the voter after they have finished entering their vote into the computer. This might be realized using a touchscreen and perhaps a special pen to reproduce the pen-and-paper voting procedure.

After the voter has entered their vote Bingo Voting and the Moran-Naor scheme begin to differ. To produce a receipt the Moran-Naor scheme needs randomness for each entry which must be entered by the voter. Also the scheme only allows for one vote per receipt, so if a voter can distribute more than one vote for each one a receipt is required. Combined with the fact that for each receipt the voter has to enter sufficient randomness for each candidate this makes the voting process (actually the receipt generation) very time consuming and will probably lead to a low acceptance. Another problem is that the voter might enter low entropy randomness when asked for many random numbers. This could compromise the security of the voting scheme. Of course a trusted random number generator could be used to generate the required randomness, but this results in other problems. A short description of this idea is found in [BMQR07a].

6.4.1.4 Bingo Voting

Like the scheme by Moran and Naor, Bingo Voting requires no special actions performed by the voter prior to the voting process, except that the voter gains access to a voting machine instead of receiving a paper ballot. The computer used by the Bingo Voting scheme to cast the vote will, after the vote was entered and confirmed by the voter, call a trusted random number generator and receives a random number for each vote of the voter. Then a receipt is printed and the voter has to compare the random numbers on the display of the trusted random number generator with the random numbers printed next to their choice(s) on the receipt. For each possible choice there must be a random number on the receipt which limits the size of the election realizable with a reasonable paper and font size for the receipt. For each vote the voter has to make one comparison which also limits the elections as each vote increases the time needed for comparison.

6.4.1.5 Comparison

In comparison to Prêt à Voter and Punchscan the voting scheme by Moran and Naor and Bingo Voting have one big advantage as they force the voter to interact with the cryptographic mechanisms of the voting scheme only *after* they have made their choice. For Bingo Voting the additional steps after the vote was cast are only necessary to ensure correctness of the vote. The scheme by Moran and Naor needs randomness as input from the voter which makes the voting process

more time consuming. As mentioned above this can be bypassed by using a random number generator as a source. In this case the minimal effort required for voting is the same as for Bingo Voting. Prêt à Voter and Punchscan both interfere with the voting process as both require special paper ballots. The voting process of Prêt à Voter requires the voter to find the candidate of their choice and mark the adjacent field. For Punchscan the voter has to find their candidate, find and remember the corresponding letter and finally find this letter and mark it with a marker.

For a voter who is concerned about correctness of the election all four voting schemes provide means to ensure correctness with additional effort. For the scheme by Moran and Naor this effort is actual part of the voting process (if randomness is entered by the voter) or optional (if a random number generator is used).

The big advantage of Moran and Naor is that their voting scheme does not need a preparation phase. Unfortunately this is bought by a cumbersome and time consuming receipt generation. Bingo Voting has the advantage that the voter only has to compare random numbers in the voting booth. The main disadvantage of Prêt à Voter and Punchscan is that both use special ballots that normal voters are not used to, may conflict with existing legal requirements and require special handling during the shredding process. Besides the random order of the candidates, Prêt à Voter is the voting scheme with the most similarities to traditional voting with paper ballots and the most flexible one.

After receiving the receipt and leaving the voting booth, the voter may check whether their receipt was published correctly. In addition to this, each voter and all auditors are able to check whether the published proofs are valid. These checks after the voting process are very similar for all four voting schemes and have been omitted in the comparison.

6.4.2 Size of Public Data

Section 5.5 describes several changes to the original Bingo Voting scheme that decrease the size of the public data.

To compare the improved scheme of Bingo Voting with the original scheme, this section gives the formulas for the size of the public data for the original scheme and the improved scheme. For the sake of completeness the increase of the size of the public data caused by prearranging dummy votes as proposed in Section 5.2 are also included in the calculations. To evaluate the improvement achieved this section also presents the size of the public data for the three scenarios described in Section 6.1.

6.4.2.1 Published Data Size of the Original Version

Bingo Voting requires one dummy vote per candidate and per voter. If a voter has more than one vote and can give more than one vote to one candidate, this increases the number of required dummy votes.

Let v be the number of voters, n the number of candidates without abstention and void, s the number of votes a single voter can distribute, and $s_{\max}$ the maximum number of votes each vote may cast for a single candidate. Then

$$d_r = n \cdot s_{\max} + 2 \cdot s$$

gives the number of random numbers on each receipt. Note that this includes the random numbers required for the ‘invalid’ and ‘abstention’ candidates. As each

random number on a receipt might be a dummy random number there, must be

$$d_t = v \cdot d_r$$

dummy random numbers (and therefore the same number of dummy votes) in total. If $\text{size}_{\text{COM}}^{\text{old}}$ is the size of a single commitment then

$$t_1^{\text{old}} = d_t \cdot 2 \cdot \text{size}_{\text{COM}}^{\text{old}}$$

is the total size of all commitments that must be published before the election. Remember that a dummy vote consists of two commitments, one containing the dummy random number and one containing the candidate.

In the original paper of Bingo Voting the voting authority proves that each candidate has the same number of dummy votes using RPC before the election. With k iterations of these proofs probability of an incorrect proof that was not detected is 2^{-k}. For each iteration, each commitment of the candidate names is rerandomized and then the rerandomized commitments are either opened or the randomness is published. For this, the new commitments plus the opening information or the rerandomization information must be published. So for each original commitment we publish one new plus one randomization factor or opening information per iteration of the RPC proof.

The total amount of published data for the RPC proof before the election therefore is

$$t_2^{\text{old}} = d_r \cdot v \cdot \left(2 \cdot \text{size}_{\text{COM}}^{\text{old}} \cdot k + 2 \cdot \text{size}_{\text{COM}}^{\text{old}}\right) \tag{6.8}$$

where k is the number of parallel instances for the Fiat-Shamir heuristic. The original version of Bingo Voting recommends $k = 80$.

After the election the voting authority publishes one proof for each receipt that it is well-formed. For this the voting authority proves that there are $d_r - s$ dummy random numbers on the receipt, that at all dummy random numbers on the receipt originate from a dummy vote for the candidate the dummy random number is associated with on the receipt. For this the voting authority proves that the content of the receipt corresponds to the content of the $d_r - s$ dummy votes used to form the receipt plus s new commitments for the fresh random numbers and the elected candidates. The total size of the proofs that each receipt is well-formed is

$$t_3^{\text{old}} = d_r \cdot v \cdot t \cdot 2 \cdot \left(2 \cdot \text{size}_{\text{COM}}^{\text{old}} \cdot k + 2 \cdot \text{size}_{\text{COM}}^{\text{old}}\right) \tag{6.9}$$

where t is the turnout of the election (so $v \cdot t$ is the number of votes cast during the election).

The last part of the published data is the opening information for the unused dummy votes. In the original scheme, the size of the data is

$$t_4^{\text{old}} = d_r \cdot v \cdot (1 - t) \cdot 2 \cdot \text{size}_{\text{COM}}^{\text{old}} + s \cdot v \cdot t \cdot 2 \cdot \text{size}_{\text{COM}}^{\text{old}}. \tag{6.10}$$

The total size of the public data for an election using the original version of Bingo Voting is

$$t^{\text{old}} = t_1^{\text{old}} + t_2^{\text{old}} + t_3^{\text{old}} + t_4^{\text{old}}. \tag{6.11}$$

6.4.2.2 Published Data Size of the Improved Version

Let v be the number of voters, n the number of candidates without abstention and void, s the number of votes a single voter can distribute, $s_{\max}$ the maximum number of votes each vote may cast for a single candidate, and t the turnout of the election. As in the previous section, with

$$d_r = n \cdot s_{\max} + 2 \cdot s$$

we denote the number of random numbers on each receipt and with

$$d_t = v \cdot d_r$$

the number of dummy votes in total. The size of the first part for the improved scheme,

$$t_1^{\text{new}} = n \cdot v \cdot 2 \cdot \text{size}_{\text{COM}}^{\text{new}}, \tag{6.12}$$

is almost identical to the size of the dummy votes for the original scheme except for the smaller commitment size $\text{size}_{\text{COM}}^{\text{new}}$.

In contrast to the original scheme, the improved scheme proves that all candidates received the same number of dummy votes during the preparation phase after the voting phase. Consequently, the proof only includes the content of the candidate part of the dummy votes actually used in the election, reducing the size

$$t_2^{\text{new}} = v \cdot t \cdot (d_r - s) \cdot 8 \cdot \text{size}_{\text{COM}}^{\text{new}} \tag{6.13}$$

substantially in comparison to the proofs of the original scheme. The exact advantage depends on the turnout, as the size of this proof if no longer depending on the number of total voters but on the number of voters that actually voted, and the number of candidates. The lower the turnout and the fewer candidates there are the larger the portion of prepared dummy votes that remain unused and are opened instead of included in the proof of equal distribution of dummy votes.

In the improved scheme, the proof that each receipt is well-formed is of size

$$t_3^{\text{new}} = d_r \cdot v \cdot t \cdot 8 \cdot \text{size}_{\text{COM}}^{\text{new}}, \tag{6.14}$$

which is significantly smaller than the proofs for each receipt in the original scheme, as only a single instance of the proof is required. The fact that each dummy vote is combined into a single commitment as described in Section 5.5.2 decreases the number of commitments as input for the proof also decrease the size.

For the improved scheme using prearranged dummy votes the opening of unused dummy votes of used clusters is more complicated, so the total size is

$$t_4^{\text{new}} = n \cdot v \cdot (1 - t) \cdot \text{size}_{\text{COM}}^{\text{new}} + v \cdot t \cdot 8 \cdot \text{size}_{\text{COM}}^{\text{new}}. \tag{6.15}$$

Since this includes the size of the proofs that are required when prearranging dummy votes, this part may even be bigger than just the decommit information for the original version. Prearranging dummy votes is not required to use the optimized proofs, but the resulting increase of the public data was included to demonstrate that the negative impact of prearranging dummy votes on the size of the public data is not significant.

The total size of the public data for an election using the improved version of Bingo Voting is

$$t^{\text{new}} = t_1^{\text{new}} + t_2^{\text{new}} + t_3^{\text{new}} + t_4^{\text{new}}. \tag{6.16}$$

	(a)	(b)	(c)
t^{old}	83 877 GB	355 272 GB	11 735 262 GB
t^{new}	723 GB	2997 GB	97 305 GB

Figure 6.3: Size of the public data for three scenarios for the original and the improved Bingo Voting scheme: (a) German Bundestag elections 2009, (b) US presidential elections 2008, (c) Indian general election 2009. This data does not include the receipts which is identical for both systems and small compared to the proofs.

6.4.2.3 Size Improvement

Figure 6.3 presents the total size of the public data of a Bingo Voting election for the three scenarios described in 6.1 and both the original an the improved scheme.

The results show that the public data of the improved scheme is about one hundred times smaller than what the original scheme would have had to publish. This improvement increases with lower turnout and fewer candidates (as only dummy votes used on a receipt are included in the proofs for the correct number of dummy votes). While the public data required for the verification of a Bingo Voting election remains large, the examples show that it is not impossible to handle the data of a Bingo Voting election with several hundred millions of voters.

6.4.3 Legal Considerations

A thorough analysis of the legal requirements for the deployment of Bingo Voting in a real-world political election is not only very specific to the country and the electoral legislation but also very hard due to a generally unclear legal situation concerning cryptographic voting schemes. Cryptographic voting schemes are normally not considered explicitly in legislation and interpreting a law is out of scope of this work.

This section briefly discusses the decision of the German Federal Constitutional Court about the deployment of voting machines during the German Federal Elections in 2005.

On March 3, 2009 the German Federal Constitutional Court (Bundesverfassungsgericht) ruled that the employment of voting machines during the German Federal Election in 2005 was unconstitutional. The ruling also stated that the German *Federal regulation for voting devices* (Bundeswahlgeräteverordnung) was insufficient to ensure elections in accordance with the German Constitution.

The press release [off09] summarizes the ruling as follows:

> "[T]he use of electronic voting machines requires that the essential steps of the voting and of the determination of the result can be examined by the citizen reliably and without any specialist knowledge of the subject."

The argumentation of the court was in short that a democratic election must be public and that using a voting device is allowed only if this requirement of publicity is met. The ruling stressed that even a voter without special knowledge must be able to verify the compliance with the principle of a democratic election.

In the wake of the ruling several European countries abandoned the use of voting devices for political elections, including the Netherlands which almost exclusively used voting machines very similar to those used in the German Federal Elections in 2005.

The open question for cryptographic voting schemes remains whether or not the cryptographic and mathematical knowledge required to verify the proofs most such schemes use is considered a "special skill". On the one hand, this knowledge is definitely not part of the everyday life of a typical voter. On the other hand, the methods used to allocate seats according to the tally in a proportional representation election is special knowledge as well. Nevertheless it is required for the verification of the result of the election.

Until a new regulation for the use of voting devices in political elections in Germany is adopted, the requirements for such devices are unclear.

7. Conclusion

The original Bingo Voting scheme possesses several weaknesses and lacks some details that have to be addressed to make it feasible and attractive for real-world elections. This work has presented several improvements and extensions to the original protocol that turn it into a practical protocol for large and complex real-world elections.

The extensions described in Section 5.1 allows voters to abstain from an election or cast invalid votes, and enable Bingo Voting to be used in elections with multiple votes per voter and several voting machines which is important for larger elections.

The improvements described in Section 5.2 reduce the probability that a corrupted voting machine is able to change a vote when a random number is drawn for a second time during an election. This allows for shorter random numbers to be used for an election and thus improves usability. Using the suggested presentation of random numbers presented in Section 5.3, the evaluation of real-world scenarios in Section 6.3.5.2 shows that random numbers represented by three groups of three characters each are sufficient for even the largest democratic election in the world.

Section 5.4 describes a possible dispute resolution procedure for a Bingo Voting election. Such a procedure is essential for real-world elections to give not only the ability to detect manipulations but also to prove them.

Section 5.5 proposes several changes to the original Bingo Voting protocol that reduce the size of the data that is published for the verification of the tally of the election. The analysis of the size of public data for different real-world scenarios presented in Section 6.4.2 shows that these changes decrease the size of the public data by a factor of more than one hundred.

Section 5.6 presents the receipt-stealing attack, an attack on almost all cryptographic voting schemes. It also presents a countermeasure that makes such an attack much harder. This countermeasure is not only applicable to Bingo Voting but to all cryptographic voting schemes that use computers to create and print the receipts.

The security analysis in Section 6.3 shows which assumptions are necessary to achieve the different properties of Bingo Voting. Section 6.4 discusses the practical aspects of the Bingo Voting scheme and shows that the size of the public data and

the length of the random numbers are both practical even for very large elections.

The question which voting scheme should be used for elections must at least partly be addressed by a political discussion. In order to be able to lead such a discussion a listing of the properties of each voting scheme as well as the assumptions required for those properties is indispensable. This work answers these questions for the case of Bingo Voting. This does not only include the security properties but also practical requirements. This work gives several improvements for usability and the size of the public data, and gives a thorough analysis for both the required length of random numbers as well as the expected size of the of the data published for the verification of the tally.

There still remain some problems and open questions that have to be addressed before Bingo Voting may be employed in a real-world election.

The most important assumption for Bingo Voting is that the trusted random number generator draws fresh numbers truly at random. Currently, the randomness is generated by a digital trusted random number generator for practical reasons, which is not convincing. A method to quickly, reliably, and, most importantly, convincingly generate randomness, display it and make it computer readable would greatly increase the verifiability of Bingo Voting.

In a complex election in which a voter has more than a single vote another problem of Bingo Voting appears. The current proofs do not address the problem that a corrupted voting machine in cooperation with a corrupted voter is able to cast a partially invalid vote. The proof for each receipt that it is well-formed does not prove that either all votes were for the invalid candidate or none was. This is an inherent property of the Bingo Voting protocol but may be fixed using the same method that allows for the ranking of candidates as described in Section 5.1.3. This method increases the size of the public data and may make Bingo Voting impractical. On the other hand, the consequences of such a behaviour may be considered non-critical. While a partially invalid ballots is impossible to generate in a paper-based election, it may also be considered a valid alternative way of voting.

Another disadvantage of using Bingo Voting for complex elections is that the receipts quickly grow large. The number of random numbers on a receipt is the number of votes a voter has, times the number of candidates. This also makes verifying the correctness of the receipt inside the voting booth laborious and time-consuming. For each vote the voter distributed they must find a specific random number on the receipt and compare it to the display of the trusted random number generator. The fact that a receipt becomes larger for more complex elections may pose a problem for all cryptographic voting schemes. The exact complexity for which a cryptographic voting scheme is practical may differ, however. A usability study to determine the maximal number of candidates and votes per voter for which Bingo Voting is still practical is currently missing.

The improvements in this work are a big and important step towards a practical application for Bingo Voting.

Bibliography

[ACvdG07] R. Araújo, R. Custódio, and J. van de Graaf. A verifiable voting protocol based on farnel. In *IAVoSS Workshop On Trustworthy Elections (WOTE2007)*, Juni 2007.

[ACvdG10] Roberto Araújo, Ricardo Custódio, and Jeroen van de Graaf. A verifiable voting protocol based on farnel. In David Chaum, Markus Jakobsson, Ronald Rivest, Peter Ryan, Josh Benaloh, Miroslaw Kutylowski, and Ben Adida, editors, *Towards Trustworthy Elections*, volume 6000 of *Lecture Notes in Computer Science*, pages 274–288. Springer Berlin / Heidelberg, 2010.

[ADC02] R. Araújo, A. Devegili, and R. Custódio. Farnel: Um protocolo criptográfico para votação digital. In *Proceedings of II Workshop em Segurança de Sistemas Computacionais, Búzios, Rio de Janeiro, Brasil*, 2002.

[Alb08] Debra Alban. Number of votes cast set record, but voter turnout percentage didn't, 2008.

[AMBS07] Joerg Arzt-Mergemeier, Willi Beiss, and Thomas Steffens. The digital voting pen at the hamburg elections 2008: Electronic voting closest to conventional voting. In Ammar Alkassar and Melanie Volkamer, editors, *E-Voting and Identity*, volume 4896 of *Lecture Notes in Computer Science*, pages 88–98. Springer Berlin / Heidelberg, 2007.

[AN06] Ben Adida and C. Andrew Neff. Ballot casting assurance. In *EVT'06: Proceedings of the USENIX/Accurate Electronic Voting Technology Workshop 2006 on Electronic Voting Technology Workshop*, pages 7–7, Berkeley, CA, USA, 2006. USENIX Association.

[AN09] Ben Adida and C. Andrew Neff. Efficient receipt-free ballot casting resistant to covert channels. In *Proceedings of the 2009 conference on Electronic voting technology/workshop on trustworthy elections*, EVT/WOTE'09, pages 11–11, Berkeley, CA, USA, 2009. USENIX Association.

[AR08a] Roberto Araújo and Peter Y. A. Ryan. Improving the farnel, three-ballot, and randell-ryan voting schemes. Cryptology ePrint Archive, Report 2008/082, 2008. `http://eprint.iacr.org/2008/082`.

[AR08b] Roberto Araújo and Peter Y. A. Ryan. Improving the farnel voting scheme. In *International Conference on Electronic Voting (EVOTE)*, 2008.

[BHK+09] Jens-Matthias Bohli, Christian Henrich, Carmen Kempka, Jörn Müller-Quade, and Stefan Röhrich. Enhancing electronic voting machines on the example of bingo voting. In *IEEE Transactions on Information Forensics and Security*, volume 4, pages 745–750, 2009.

[BHM08] Michael Backes, Catalin Hritcu, and Matteo Maffei. Automated verification of remote electronic voting protocols in the applied pi-calculus. *Computer Security Foundations Symposium, IEEE*, 0:195–209, 2008.

[BHMQ+08] Michael Bär, Christian Henrich, Jörn Müller-Quade, Stefan Röhrich, and Carmen Stüber. Real world experiences with bingo voting and a comparison of usability. In *Workshop on Trustworthy Elections (WOTE)*, 2008.

[BLS+03] Benjamin B. Bederson, Bongshin Lee, Robert M. Sherman, Paul S. Herrnson, and Richard G. Niemi. Electronic voting system usability issues. In *CHI '03: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 145–152, New York, NY, USA, 2003. ACM.

[BMQR07a] Jens-Matthias Bohli, Jörn Müller-Quade, and Stefan Röhrich. Bingo voting: Secure and coercion-free voting using a trusted random number generator. In A. Alkassar and M. Volkamer, editors, *VOTE-ID 2007*, volume 4896 of *Lecture Notes in Computer Science*, pages 111–124. Springer-Verlag, 2007.

[BMQR07b] Jens-Matthias Bohli, Jörn Müller-Quade, and Stefan Röhrich. Bingo voting: Secure and coercion-free voting using a trusted random number generator. Cryptology ePrint Archive, Report 2007/162, 2007.

[Bär08] Michael Bär. *Analyse und Vergleich verifizierbarer Wahlverfahren.* Diploma thesis, Universität Karlsruhe (TH), May 2008.

[BT94] Josh Benaloh and Dwight Tuinstra. Receipt-free secret-ballot elections (extended abstract). In *Proceedings of the twenty-sixth annual ACM symposium on Theory of computing*, STOC '94, pages 544–553, New York, NY, USA, 1994. ACM.

[Bun09] Der Bundeswahlleiter. Endgültiges ergebnis der bundestagswahl 2009 (final result of the german bundestag election 2009), 2009.

[Can00] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. Cryptology ePrint Archive, Report 2000/067, December 2000.

[Can01] R. Canetti. Universally composable security: a new paradigm for cryptographic protocols. In *Foundations of Computer Science, 2001. Proceedings. 42nd IEEE Symposium on*, pages 136 – 145, oct. 2001.

[CCC+08] David Chaum, Richard Carback, Jeremy Clark, Aleksander Essex, Stefan Popoveniuc, Ronald L. Rivest, Peter Y. A. Ryan, Emily Shen, and Alan T. Sherman. Scantegrity ii: End-to-end verifiability for optical scan election systems using invisible ink confirmation. In *Proceedings of the conference on Electronic voting technology*, EVT'08, pages 14:1–14:13, Berkeley, CA, USA, 2008. USENIX Association.

[CCC+10] Richard T Carback, David Chaum, Jeremy Clark, John Conway, Aleksander Essex, Paul S. Herrnson, Travis Mayberry, Stefan Popoveniuc, Ronald L. Rivest, Emily Shen, Alan T. Sherman, and Poorvi L. Vora. Scantegrity ii municipal election at takoma park: The first e2e binding governmental election with ballot privacy. Proceedings of the 19th USENIX Security Symposium, 2010.

[CDvdG87] David Chaum, Ivan Damgård, and Jeroen van de Graaf. Multiparty computations ensuring privacy of each party's input and correctness of the result. In *Advances in Cryptology – CRYPTO '87*, volume 293 of *Lecture Notes in Computer Science*. Springer, 1987.

[CEC+08] D. Chaum, A. Essex, R. Carback, J. Clark, S. Popoveniuc, A. Sherman, and P. Vora. Scantegrity: End-to-end voter-verifiable optical-scan voting. *Security Privacy, IEEE*, 6(3):40 –46, may-june 2008.

[CF85] Josh D. Cohen and Michael J. Fischer. A robust and verifiable cryptographically secure election scheme. In *Proceedings of the 26th Annual Symposium on Foundations of Computer Science*, pages 372–382, Washington, DC, USA, 1985. IEEE Computer Society.

[CH10] Jeremy Clark and Urs Hengartner. On the use of financial data as a random beacon. In *Proceedings of the 2010 international conference on Electronic voting technology/workshop on trustworthy elections*, EVT/WOTE'10, pages 1–8, Berkeley, CA, USA, 2010. USENIX Association.

[Cha86] David Chaum. Showing credentials without identification: Signatures transferred between unconditionally unlinkable pseudonyms. In Franz Pichler, editor, *Advances in Cryptology – EUROCRYPT' 85*, volume 219 of *Lecture Notes in Computer Science*, pages 241–244. Springer Berlin / Heidelberg, 1986.

[Cha02] David Chaum. Secret-ballot receipts and transparent integrity: Better and less-costly electronic voting at polling places. Technical Report, VReport, 2002.

[Cha04] David Chaum. Secret-ballot receipts: True voter-verifiable elections. *IEEE Security and Privacy*, 2(1):38–47, Januar-Februar 2004.

[CRS04] David Chaum, Peter Ryan, and Steve Schneider. A practical voter-verifiable election scheme. Technical Report CS-TR:880, University of Newcastle, Dezember 2004.

[CRS05] David Chaum, Peter Ryan, and Steve Schneider. A practical voter-verifiable election scheme. In Sabrina De Capitani di Vimercati, Paul Syverson, and Dieter Gollmann, editors, *Computer Security – ESORICS 2005*, volume 3679 of *Lecture Notes in Computer Science*, pages 118–139. Springer, 2005.

[Cus01] Ricardo Custódio. Farnel: um protocolo de votação papel com verificabilidade parcial. Invited Talk at Simpósio Segurança em Informática (SSI), November 2001.

[Dan08] Hadmut Danisch. Kritik an Bingo Voting, 2008. at time of writing available at `http://www.danisch.de/dok/BingoKritik.pdf`.

[DJ01] Ivan Damgård and Mads Jurik. A generalisation, a simplification and some applications of paillier's probabilistic public-key system. In Kwangjo Kim, editor, *Public Key Cryptography*, volume 1992 of *Lecture Notes in Computer Science*, pages 119–136. Springer Berlin / Heidelberg, 2001.

[ECCP07] Aleks Essex, Jeremy Clark, Rick Carback, and Stefan Popoveniuc. Punchscan in practice: An e2e election case study. In *IAVoSS Workshop on Trustworthy Elections (WOTE 2007)*, 2007.

[EHH11] Aleksander Essex, Christian Henrich, and Urs Hengartner. Single layer optical-scan voting with fully distributed trust. Cryptology ePrint Archive, Report 2011/568, 2011.

[FDBV11] Richard Frankland, Denise Demirel, Jurlind Budurushi, and Melanie Volkamer. Side-channels and evoting machine security: Identifying vulnerabilities and defining requirements. In *International Workshop on Requirements Engineering for Electronic Voting Systems (REVOTE 2011)*, pages 37 – 46, 2011.

[FS87] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew Odlyzko, editor, *Advances in Cryptology – CRYPTO' 86*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer Berlin / Heidelberg, 1987.

[GB07] Stephen N. Goggin and Michael D. Byrne. An examination of the auditability of voter verified paper audit trail (vvpat) ballots. In *Proceedings of the USENIX Workshop on Accurate Electronic Voting Technology*, EVT'07, pages 10–10, Berkeley, CA, USA, 2007. USENIX Association.

[Gol01] Oded Goldreich. *Foundations of Cryptography*, volume Basic Tools. Cambridge University Press, 2001.

[Gro02] Jens Groth. A verifiable secret shuffe of homomorphic encryptions. In Yvo Desmedt, editor, *Public Key Cryptography – PKC 2003*, volume 2567 of *Lecture Notes in Computer Science*, pages 145–160. Springer Berlin / Heidelberg, 2002.

[Gro10] Jens Groth. A verifiable secret shuffle of homomorphic encryptions. *Journal of Cryptology*, 23:546–579, 2010.

[HNH+07] Paul S. Herrnson, Richard G. Niemi, Michael J. Hanmer, Benjamin B. Bederson, Frederick G. Conrad, and Michael Traugott. The not so simple act of voting: An examination of voter errors with electronic voting, 2007.

[JCJ05] Ari Juels, Dario Catalano, and Markus Jakobsson. Coercion-resistant electronic elections. In *WPES '05: Proceedings of the 2005 ACM workshop on Privacy in the electronic society*, pages 61–70, New York, NY, USA, 2005. ACM.

[Joh28] J. B. Johnson. Thermal agitation of electricity in conductors. *Phys. Rev.*, 32:97–109, Jul 1928.

[Jon05] Douglas W. Jones. Chain voting, August 2005.

[KKW06] Aggelos Kiayias, Michael Korman, and David Walluck. An internet voting system supporting user privacy. In *ACSAC '06: Proceedings of the 22nd Annual Computer Security Applications Conference*, pages 165–174, Washington, DC, USA, 2006. IEEE Computer Society.

[Kob87] Neal Koblitz. Elliptic curve cryptosystems. *Mathematics of Computation*, 48(177):203 – 209, 1987.

[KR05] Steve Kremer and Mark Ryan. Analysis of an electronic voting protocol in the applied pi calculus. In Mooly Sagiv, editor, *Programming Languages and Systems*, volume 3444 of *Lecture Notes in Computer Science*, pages 140–140. Springer Berlin / Heidelberg, 2005.

[KRS10] Steve Kremer, Mark Ryan, and Ben Smyth. Election verifiability in electronic voting protocols. In Dimitris Gritzalis, Bart Preneel, and Marianthi Theoharidou, editors, *Computer Security – ESORICS 2010*, volume 6345 of *Lecture Notes in Computer Science*, pages 389–404. Springer Berlin / Heidelberg, 2010.

[KSW05] Chris Karlof, Naveen Sastry, and David Wagner. Cryptographic voting protocols: a systems perspective. In *SSYM'05: Proceedings of the 14th conference on USENIX Security Symposium*, pages 3–3, Berkeley, CA, USA, 2005. USENIX Association.

[KTV09] Ralf Küsters, Tomasz Truderung, and Andreas Vogt. A game-based definition of coercion-resistance and its applications. Cryptology ePrint Archive, Report 2009/582, 2009.

[KTV11] Ralf Küsters, Tomasz Truderung, and Andreas Vogt. Verifiability, privacy, and coercion-resistance: New insights from a case study. Cryptology ePrint Archive, Report 2011/517, 2011.

[LPS08] Yehuda Lindell, Benny Pinkas, and Nigel Smart. Implementing two-party computation efficiently with security against malicious adversaries. In Rafail Ostrovsky, Roberto De Prisco, and Ivan Visconti, editors, *Security and Cryptography for Networks*, volume 5229 of *Lecture Notes in Computer Science*, pages 2–20. Springer Berlin / Heidelberg, 2008.

[LR08] David Lundin and Peter Y. Ryan. Human readable paper verification of Prêt à Voter. In *ESORICS '08: Proceedings of the 13th European Symposium on Research in Computer Security*, pages 379–395, Berlin, Heidelberg, 2008. Springer-Verlag.

[Mil86] Victor Miller. Use of elliptic curves in cryptography. In Hugh Williams, editor, *Advances in Cryptology – CRYPTO '85 Proceedings*, volume 218 of *Lecture Notes in Computer Science*, pages 417–426. Springer Berlin / Heidelberg, 1986.

[MN06] Tal Moran and Moni Naor. Receipt-free universally-verifiable voting with everlasting privacy. In Cynthia Dwork, editor, *Advances in Cryptology - CRYPTO 2006*, volume 4117 of *Lecture Notes in Computer Science*, pages 373–392. Springer Berlin / Heidelberg, 2006.

[Nef04] C. Andrew Neff. Practical high certainty intent verification for encrypted votes. Draft at `http://www.votehere.net/vhti/documentation/vsv-2.0.3638.pdf`, 2004.

[NS95] Moni Naor and Adi Shamir. Visual cryptography. In Alfredo De Santis, editor, *Advances in Cryptology – EUROCRYPT'94*, volume 950 of *Lecture Notes in Computer Science*, pages 1–12. Springer Berlin / Heidelberg, 1995.

[Nyq28] H. Nyquist. Thermal agitation of electric charge in conductors. *Phys. Rev.*, 32:110–113, Jul 1928.

[off09] Federal Constitutional Court Press office. Use of voting computers in 2005 bundestag election unconstitutional. Press release no. 19/2009 of 3 March 2009, March 2009.

[Oka98] Tatsuaki Okamoto. Receipt-free electronic voting schemes for large scale elections. In Bruce Christianson, Bruno Crispo, Mark Lomas, and Michael Roe, editors, *Security Protocols*, volume 1361 of *Lecture Notes in Computer Science*, pages 25–35. Springer Berlin / Heidelberg, 1998. 10.1007/BFb0028157.

[Ond09] B. Ondrisek. E-voting system security optimization. In *System Sciences, 2009. HICSS '09. 42nd Hawaii International Conference on*, pages 1 –8, jan. 2009.

[Ped92] Torben Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In Joan Feigenbaum, editor, *Advances in Cryptology - CRYPTO '91*, volume 576 of *Lecture Notes in Computer Science*, pages 129–140. Springer Berlin / Heidelberg, 1992.

[PH06a] Stefan Popoveniuc and Ben Hosp. An introduction to punchscan. In *Workshop on Trustworthy Elections (WOTE)*, October 2006.

[PH06b] Stefan Popoveniuc and Ben Hosp. An introduction to punchscan. Threat Analyses for Voting System Categories, A Workshop on Rating Voting Methods, VSRW 06, June 2006.

[PH10] Stefan Popoveniuc and Ben Hosp. An introduction to punchscan. In David Chaum, Markus Jakobsson, Ronald Rivest, Peter Ryan, Josh Benaloh, Miroslaw Kutylowski, and Ben Adida, editors, *Towards Trustworthy Elections*, volume 6000 of *Lecture Notes in Computer Science*, pages 242–259. Springer Berlin / Heidelberg, 2010.

[Rab83] Michael O. Rabin. Transaction protection by beacons. *Journal of Computer and System Sciences*, 27:256–267, 1983.

[Riv06] Ronald L. Rivest. The threeballot voting system, October 2006. Draft online available at `http://people.csail.mit.edu/rivest/Rivest-TheThreeBallotVotingSystem.pdf`.

[Riv08] Ronald L. Rivest. On the notion of 'software independence' in voting systems. *Philosophical Transactions of the Royal Society A*, 366(1881):3759–3767, 2008.

[RP05] Peter Ryan and Thea Peacock. Prêt à voter: a systems perspective. Technical Report CS-TR:929, University of Newcastle, September 2005.

[RS07] Ronald L. Rivest and Warren D. Smith. Three voting protocols: Threeballot, vav, and twin. In *Proceedings of the USENIX Workshop on Accurate Electronic Voting Technology*, EVT'07, pages 16–16, Berkeley, CA, USA, 2007. USENIX Association.

[Rya69] Joanna Ryan. Grouping and short-term memory: Different means and patterns of grouping. The Quaterly Journal of Experimental Psychology 21, p.137-147, 1969.

[Rya06] P. Y. A. Ryan. Verified encrypted paper audit trails. Technical Report Series, University of Newcastle, CS-TR:966, 2006.

[Sal88] R. G. Saltman. Accuracy, integrity and security in computerized vote-tallying. *Commun. ACM*, 31(10):1184–1191, October 1988.

[SDW08] Daniel Sandler, Kyle Derr, and Dan S. Wallach. Votebox: a tamper-evident, verifiable electronic voting system. In *SS'08: Proceedings of the 17th conference on Security symposium*, pages 349–364, Berkeley, CA, USA, 2008. USENIX Association.

[Ser98] Gadiel Seroussi. Compact representation of elliptic curve points over f2n, 1998.

[Sid10] Julian Siddle. Us scientists 'hack' india electronic voting machines, 2010.

[TH03] Michael Tomz and Robert P. Van Houweling. How does voting equipment affect the racial gap in voided ballots? *American Journal of Political Science*, 47(1):46–60, 2003.

[Tha09] Shashi Tharoor. The Recurring Miracle of Indian Democracy, 2009.

[THP+05] Michael W. Traugott, Michael J. Hanmer, Won-Ho Park, Paul S. Herrnson, Richard G. Niemi, Ben B. Bederson, and Frederick G. Conrad. The impact of voting systems on residual votes, incomplete ballots, and other measures of voting behavior, 2005.

[UMQ10] Dominique Unruh and Jörn Müller-Quade. Universally composable incoercibility. In Tal Rabin, editor, *Advances in Cryptology – CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 411–428. Springer Berlin / Heidelberg, 2010.

[Uni08] American University. African-americans, anger, fear and youth propel turnout to highest level since 1964. published by Media Relations, 2008. Press Release.

[vdG09] Jeroen van de Graaf. Voting with unconditional privacy: Cfsy for booth voting. Cryptology ePrint Archive, Report 2009/574, 2009.

[Vol09] Melanie Volkamer. Implementations of electronic voting. In Wil Aalst, John Mylopoulos, Michael Rosemann, Michael J. Shaw, and Clemens Szyperski, editors, *Evaluation of Electronic Voting*, volume 30 of *Lecture Notes in Business Information Processing*, pages 13–35. Springer Berlin Heidelberg, 2009.

[WH09] Janna-Lynn Weber and Urs Hengartner. Usability study of the open audit voting system helios, 2009.

[Wic64] Wayne A. Wickelgren. Size of rehearsal group and short-term memory. *Journal of Experimental Psychology*, 68(4):413–419, 1964.