Towards Trustworthiness without Trusted Authorities

A THESIS SUBMITTED TO
THE SCIENCE AND ENGINEERING FACULTY
OF QUEENSLAND UNIVERSITY OF TECHNOLOGY
IN FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Thomas Edmund Haines

School of Electrical Engineering and Computer Science
Science and Engineering Faculty
Queensland University of Technology

2017

Copyright in Relation to This Thesis

Statement of Original Authorship

The work contained in this thesis has not been previously submitted to meet requirements for an award at this or any other higher education institution. To the best of my knowledge and belief, the thesis contains no material previously published or written by another person except where due reference is made.

Signature:

Date: 25/9/2017

To my grandmother

Abstract

Our democratic way of life is under threat. The increased computerisation and centralisation of our elections, commerce, and information poses unprecedented threats to the foundations of our society. While significant cryptographic work has been done to reduce these threats, attacks and vulnerabilities are widespread.

Often attacks occur on systems whose security has been 'proven'. Overly generous trust assumptions in the security analysis are a significant reason for this. The process of reducing these trust assumptions is non-trivial but highly necessary.

We focused on reducing reliance on trusted authorities in cryptographic protocols. The primary focus, in this broad area, was the powerful authorities which are prevalent in modern end-to-end verifiable electronic voting schemes. We contributed knowledge about how to improve public consensus systems; such as elections, commerce and polling. More specifically, we investigated how new cryptographic primitives and human involvement in the cryptographic protocols can be used to remove or reduce trusted authorities.

Our work has been both practical and theoretical. On the practical side, modifications to the in-polling booth electronic voting scheme "Prêt à Voter"—trialled in the Australian state of Victoria—have been examined. These modifications allow higher privacy without impacting useability unnecessarily. We further introduced a new online voting scheme called "VOTOR" which, among other desirable properties, prevents the authorities from learning or casting votes. On the theoretical side, we introduced a new unconditionally anonymous linkable ring signature scheme which is the first to be forward secure.

Keywords

Public Consensus Systems, Electronic Voting, Trusted Authorities, Device Independence, Authority Independence, Democracy, Online Voting, In-Polling-Booth, Elections, Ring Signature, Forward Security, Linkability, Prêt à Voter, Government

Acknowledgments

There are many people who deserve acknowledgement for their contribution and encouragement.

I would like to thank my supervisor A/Prof. Xavier Boyen, my associate supervisor Prof. Colin Fidge, and my external supervisor Dr. Vanessa Teague for their many helpful and stimulating discussions and valued support throughout my candidature.

I would also like to thank the many members of the Information Security Discipline for their support and encouragement; Dr. Ken Radke, Dr. Ernest Foo, Dr. Douglas Stebila, Prof. Colin Boyd, Prof. Josef Pieprzyk, Dr. Harry Bartlet, Dr. Leonie Simpson, Dr. Chai Wen Chuah, Dr. Janaka Alawatugoda, Raphael Amoah, James Akande, Ben Dowling, Qinyi Li, Udyani Hearth, David Myers, Nishchal Kush, Jack Parry, Chris Carr, Nicholas Rodofile, and others.

I would like to thank Douglas Wikström and the reviewers for their helpful comments on the paper [HB16a]—on which Chapter 3 is based.

My parents and brother. Their support and encouragement has been tremendous and ongoing. I am forever in your debt.

And finally to my wife. Thank you for putting up with a strange husband, whose brain was often off thinking about other things. Submitting a paper—on our honeymoon—was at least an honest beginning.

Preface

A version of Chapter 3 has been published at the 12th International Joint Conference on Electronic Voting, [HB16a]. I was the lead investigator, responsible for all major areas of concept formation, data collection and analysis, as well as manuscript composition. Xavier Boyen was the supervisory author on this project and was involved throughout the project in concept formation and manuscript composition.

A version of Chapter 4 has been published at the Australasian Information Security Conference 2016 [HB16b]. I was the lead investigator, responsible for all major areas of concept formation, data collection and analysis, as well as the majority of manuscript composition. Xavier Boyen was the supervisory author on this project and was involved throughout the project in concept formation and manuscript composition.

The material in Chapter 5 was joint-work with Xavier Boyen. I was the lead investigator, responsible for all major areas of concept formation, data collection and analysis, as well as the majority of manuscript composition. Xavier Boyen was the supervisory author on this project and was involved throughout the project in concept formation and manuscript composition.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

> *It has been said that democracy is the worst form of government except all the others that have been tried.*
>
> (Winston Churchill)

We live in a profoundly digital age. Much of our commerce, communication, civil society and cat photos are dependent on technology. This has unleashed possibilities for remote, large scale, and low cost attacks without parallel. The routine hacking of trusted organisations releases information causing financial harm, enabling identify theft and undermining our social contracts. Despite repeated outrage, there seems no end in sight to these attacks.

The recent acceleration in the adoption of new digital systems for government elections is alarming: opening elections to the same kind of attacks witnessed routinely against other organisations. The high stake nature of elections attracts powerful nation state adversaries whose power must not be underestimated. And as we have seen in the recent United States election, a winning candidate has a strong interest in maintaining trust in their election—even if there is evidence of attacks.

The effort required to adequately secure digital systems—which are notoriously complex, relying on numerous technological and human factors—is enormous. The aim of the research conducted in this PhD was to investigate how to lower or remove reliance on various 'trusted' authorities from within these digital systems. Such systems are normally technological contraptions owned and maintained by trusted human organisations. The removal of compulsorily trusted authorities reduces the number of avenues for attack, especially by the 'trusted' authority themselves. The removal of unwarranted trust increases the reliability that the system will act

in a honest and forthright way. Trust is often mandated and abused; trustworthiness, however, is earned. The title "Towards Trustworthiness without Trusted Authorities" evokes this notion.

1.1 Electronic Voting

Since much of this work focuses on electronic voting systems, we here give a brief description of electronic voting. We draw particular attention to the areas which are not commonly known.

Electronic voting systems is here used to refer to any voting system with electronic components, even if significant parts are still physical. This definition is important as it emphasises that many systems, including Australia's state and federal elections, are in fact at least partially electronic even if this is not clear to the voters. A subset of electronic voting is normally discussed within the cryptographic community; this subset of schemes can be analysed formally to prove their security properties under certain assumptions. A subset of the subset, which has nevertheless come to dominate much of the research due to its desirable properties, consists of end-to-end verifiable electronic voting schemes. The voting schemes presented in Chapter 3 and 4 are end-to-end verifiable.

End-to-end verifiable schemes allow verification of the result from end-to-end, that is, it is possible to verify that the ballots entered for casting, and only those ballots, were counted. This process is generally broken into three stages; 'Cast as Intended', 'Collected as Cast', and 'Counted as Collected'. Each step can either be universally verifiable or individually verifiable, depending on whether anyone can verify all votes or an individual can verify their own vote.

The literature of cryptographic electronic voting emerges from the mixing of voting and cryptographic literature. On the cryptographic side, Andrew Yao's landmark paper "Protocols for Secure Computations" created the field of secure multiparty computation and established voting as an often cited application of cryptography. Multiparty computation allows multiple participants to calculate a function on their inputs without revealing their inputs to each other. There is a clear parallel to voting. However, multiparty computation requires the parties to engage in a complicated process which fails to achieve several natural properties of voting, and scales very poorly to large numbers of participants.

The slow re-introduction of core voting ideas into cryptographic voting has brought the field to a point where useability is the highest good, and the security goals are defined by French

Enlightenment and Chartism ideals. This focus has lead to weaker than ideal authority and device independence.

The following list introduces, informally, many of the desirable properties in electronic voting.

1. Privacy: No information about an individual's vote can be determined by another party.
2. Receipt-Freeness: No individual can prove information about their vote.
3. Democracy:
   (a) Eligibility: Only eligible voters should be able to vote.
   (b) Uniqueness: Eligible voters can vote at most once.
4. Correctness:
   (a) 'Cast as Intended': Votes are cast as the voter intended.
   (b) 'Collected as Cast': Votes are collected as they were cast.
   (c) 'Counted as Collected': Votes are counted as they were collected.
5. Verifiability:
   (a) Individual Verifiability: Individuals can verify that their vote was counted.
   (b) Universal Verifiability: Everyone can verify that the election or some stage thereof was performed correctly.
6. Auditability: The system's correctness can be verified and misbehaviour proven.
7. Reliability: The election scheme should not be easy to block from running.
8. Useability: All members of the electorate should be able to vote conveniently.

The security properties of electronic voting are hard to achieve—even assuming a secure voting environment—-since the voter themself becomes an adversary. The useability constraints, also, should not be underestimated. "**All** members of the electorate" includes, in most cases, a huge diversity of socio-economic situations, backgrounds, and languages—not to mention disabilities of cognition, vision and hearing. The combination of the high security and useability

requirements make the construction of voting schemes—with even most desirable properties—a demanding task.

A further issue is the social acceptance of any new electronic scheme. Several demographics, notably politicians and those unfamiliar with an electronic system, would be adverse to the adaption of a new system. The advances detailed in this thesis are unified in increasing the value of electronic voting and hence reducing aversion to it.

1.2 Objectives and Outcomes

The main objectives of this thesis are listed below:

1. **To contribute new methods to involve humans in cryptographic protocols.**

   Cryptography generally assumes that humans have access to computational devices which are completely trusted. This is necessary to complete the complicated mathematical calculations cryptography involves. However, home computers—and to a lesser degree, corporate and government—are rarely secure, particularly against targeted attacks. Furthermore, in a government election, trusting a provided device amounts to placing enormous trust in the authority's character and abilities, which we have already argued is unwise. It is, therefore, of interest to investigate how greater human involvement can reduce reliance on devices and authorities.

2. **To apply these new methods to voting to remove reliance on trusted authorities.**

   The new methods mentioned above, while of independent interest, behove application. We, therefore, aim to apply these methods to existing electronic voting schemes, preferably in use, to reduce reliance on trusted authorities.

3. **To determine which recent cryptographic primitives are well suited to voting.**

   There has been an ongoing development of powerful cryptographic primitives—expressive encryption and signature schemes to name a few—without a corresponding adoption in electronic voting. Instead, complicated voting procedures have been preferred to achieve the desired properties. The right choice of cryptographic primitives may usher in simpler voting schemes with equivalent or stronger security. However, some of these primitives—including some which claim voting as an application—are actually not suitable for voting,

since their security model is too weak. It is, therefore, a non-trivial and interesting task to investigate which recent cryptographic primitives are well suited to voting, as part of a whole system.

4. **To develop new cryptographic primitives which enable stronger and simpler voting schemes.**

   A study of voting schemes often turns up desirable properties which could be added to existing schemes. However, it is often cumbersome to adjust the protocol to achieve these properties—which has effects on useability. Many of these properties could be achieved by the primitives used, and simply inherited by the voting scheme without any adjustment to the protocol. Investigating how to improve these primitives therefore assists our main goals.

5. **To use these recent and new cryptographic primitives to improve existing voting schemes.**

   Applying the new and recent cryptographic primitives from objectives 3 and 4 to existing voting schemes is not only an interesting exercise but also validates further the applicability of the primitives.

We now discuss the outcomes in relation to these objectives.

In this thesis we took an exploratory approach targeting practical improvement to systems used for important governmental and democratic functions. This involved significant theoretical research as well as prototyping. Several new schemes and variants of existing schemes were presented which offer significant advantages over previous work.

With respect to objective 1, we made an extensive study of existing human cryptography. This allowed us to ascertain the advantages, limitations and viable combinations of existing work. We presented two new methods to involve humans in cryptographic protocols. Chapter 3 details these methods and their cryptographic properties.

We made a careful study of existing end-to-end verifiable voting schemes currently being used in government elections. We further investigated how the human cryptographic methods, studied and created in the course of completing objective 1, could be used to improve the schemes. Chapter 3 then details how these methods can be applied to the voting system Prêt à Voter. The resulting scheme is less reliant on a trusted authority, achieving objective 2.

Objective 3 was satisfied by an extensive literature review, followed by an in-depth analysis

of promising schemes. The results can be found in Chapters 2, 3 and 4.

The aim of objective 4 was achieved by creating a new linkable ring signature which provides forward security. Chapter 5 details the results.

Objective 5 was achieved by further exploration, based on the knowledge gained from the successful completion of the previous objectives. We presented a new Internet voting scheme with desirable properties. The details of objective 5's completion can be found in Chapter 4.

1.3 Structure

This thesis is divided into six chapters, including this introductory chapter.

Chapter 2 is the literature review. It provides a background on the requirements of voting and a brief historical overview. It progresses with an overview of cryptography related to voting and importantly why cryptography is useful for voting. The chapter then introduces cryptographic voting schemes and discusses the most prominent ones in some detail. Finally, it presents the open problems in the field and gives a conclusion.

Chapter 3—entitled "Truly Multi-Authority *'Prêt-à-Voter'*"—contains work on removing trusted devices from the Prêt à Voter voting system. The chapter introduces the problem and solution before giving an in-depth overview of Prêt à Voter. It then presents the details of the solution. Next the voter journey of the two variants is introduced. Finally, the chapter discusses practical matters of running such an election before summarising the work.

Chapter 4—entitled "VOTOR: Conceptually Simple Remote Voting against Tiny Tyrants"—contains a new online voting system which relies less on trusted authorities than comparable schemes. The chapter opens by introducing the problem and intuition for the solution. It then goes into preliminaries before presenting the details of the construction. The security assumptions and analysis follow. Comparison and discussion precede the conclusion.

Chapter 5—entitled "Forward Secure Linkable Ring Signatures"—presents a new candidate strategy for a linkable ring signature which provides forward security, under an assumption (currently debated in cryptographic circles) that certain multilinear maps exist. The chapter opens with an introduction before going into the security definitions. It provides a summary of multilinear maps including major issues. The construction then follows. Correctness and security analyses are presented. A combinatorial generalisation is then presented. A conclusion

rounds off the chapter.

Chapter 6 concludes the thesis and summarises the work presented. It also provides recommendations on the use of electronic voting in government elections and future research directions.

Chapter 2

Literature Review

> *The first principle of a free society is an untrammeled flow of words in an open forum.*
>
> (Adlai E. Stevenson)

It is necessary to limit the scope of the literature review in order to achieve some measure of clarity and brevity. For this reason, we largely focus on the issues present in electronic voting, rather than wider areas of public consensus systems.

2.1 Introduction

Voting has a long history of use in facilitating group decision making. However, it is with the democratic process that it is now most strongly associated. It is not surprising then that it is with the Greeks of antiquity, whom many consider the fathers of democracy, that we see machines first being used to assist with voting [Bru66].

Unfortunately, mechanical and electronic voting, on their own, rely upon the correctness of the machines used. For this reason they fell out of popularity until the 1960s, when they resurfaced in the United States. Since then there have been a large number of new machines including punch cards, optical scan systems and DRE (Direct-Recording Electronic) voting machines. A major advantage of machine voting over papers ballots is increased useability, particularly for disabled voters [BLS+03]. The correctness concerns are still present in these systems, which leads us to the use of cryptography in voting.

In government elections, in particular, electronic voting in one form or another, has seen

widespread use. Proponents have, and continue to, argue that it increases efficiency, encourages participation and turnout, reduces queues, and speeds up counting. However, as we noted in the introduction, the useability and security requirements are not easy to meet simultaneously. There has naturally, therefore, been ongoing research interest into how to secure electronic voting.

Research into electronic voting has undergone a series of paradigm shifts. We will take a chronological view of the literature in this field and divide it into sections following the evolution of various approaches We will present the questions and the solutions through which the field has travelled. We will then summarise the current state of the field and highlight several existing research gaps. Finally, we will discus what kind of contribution to knowledge a solution would make.

2.1.1 Structure

The field of electronic voting is complicated, multidisciplinary and highly interconnected. Insofar as we focus on end-to-end verifiable electronic voting, we are in truth discussing two topics. The first is electronic voting, which is the use of machines for the casting or collection of ballots. The second is cryptography, which allows us to extract correctness guarantees without breaking privacy. We structure our literature review along this divide.

The literature review is structured as follows:

Requirements of Voting We begin by discussing the requirements of voting as they have emerged in the modern democratic process. This is necessary since any attempt to solve the real world problems in secure voting must be anchored in actual requirements.

Cryptography Related to Voting We present a chronological view of the cryptographic research on practical and secure voting systems. We discuss human cryptography and its applications to voting. We conclude with a summary of why cryptography is powerful for voting, with specific emphasis on the guarantees that can be extracted using cryptography beyond secret communication.

Cryptographic Voting Schemes We present a brief overview of the literature of cryptographic voting schemes, dividing schemes into sections based on the primitives used.

Current State of the Art We provide a topical overview of current electronic voting schemes; this is followed by a more in depth analysis of several prominent schemes.

Open Problem(s) We describe a model of adversarial powers for voting, which combines with the requirements of voting to define a threat model. This threat model allows us to make clear the open problems we wish to solve. We also briefly point out the contradictions between various key properties.

Conclusion We summarise all the information presented.

2.2 Requirements of Voting

The most prevalent security issue present in voting, aside from correctness, is immunity to coercion. This generally takes the form of predatory coercion where a stronger party attempts to align another's actions to best suit their interests. Democracy, "rule of the people", aims to shift political power to the people. For a democracy to work, power must be left with the people, not transferred to a powerful minority through coercion.

Elections were used throughout antiquity and the medieval period to select rulers. Generally during this period voter coercion was expected, and indeed the elections attempted to discern who had the most power. As the groups who were allowed to vote increased in size and the concept of "one man, one vote" became more prevalent, voter coercion was seen as a more significant issue. The French Constitution of 1795[1] stated "All elections are to be held by secret ballot", in an attempt to ensure coercion was mitigated. This has since remained the primary tool against coercion.

Chartism was a political reform movement of the mid 19th century. It occurred largely in Britain but had effects as far away as Australia. Many of the reforms called for by the movement are familiar to us: equal representation, paid politicians, secret ballot and suffrage without property. At a higher level, the following properties were promoted:

1. Universal Suffrage: All citizens meeting a minimal set of requirements should be allowed and enabled to vote.

[1] For an extended overview see [CC07].

2. Freedom of Representation: Any member of the electorate with sufficient support should be able to hold office.

3. Freedom from Coercion: Voters should not suffer persecution for voting.

4. Equal Representation: Each vote should have roughly the same weight.

Philosophically, the author espouses the movements mentioned above, which assume that compulsorily secret ballots are strongly beneficial—if not necessary—for a healthy democracy. This position, which is by no means universally held, informs many of the later discussions on electronic voting.

In the modern era, there has since been extensive research into group decision making and social choice theory. This research has aimed to provide methods which combine group preferences into group decisions without paradoxes. The research has also revealed many impossibility results—Arrow's theorem [Arr50] being perhaps the most famous, which says "That no voting system can respect the Unanimity, Independence of irrelevant candidates without being a Dictatorship". We do not deal with the issues that dominate that field. Rather, we aim to securely realise the properties required of an implementation of a social choice method which that field views as best. Fortunately most electronic voting system are agnostic as to the underlying social choice function. Nevertheless, we think it useful to highlight a list of the key properties here:

1. Anonymity: The identity of the voters should not affect the impact of their vote.

2. Neutrality: The names of the candidates, or the options, does not effect the outcome.

3. Universal Domain: Voters are free to have, and express, any preference order over the candidates.

4. Non-dictatorship: The function does not mimic the preferences of a single voter.

5. Independence of Irrelevant Alternatives: The existence of a third candidate should not effect the pairwise preference of two candidates in the outcome if the voters' pairwise preference of the two candidates remains the same.

6. Monotonicity: An option should not be hurt by the voter placing it higher.

7. Unanimity: If every voter prefers one option over another, then so must the social choice function.

2.2.1 Useability

Useability is among the most important properties of a voting scheme. It follows directly from the principal of universal suffrage. If we want all citizens with a minimal set of requirements to be able to vote, the scheme should also have a minimal set of requirements for use. Useability becomes enormously more difficult when you consider the mix of language and cultural backgrounds, ages and demographics of the electorate. Useability can be broken down into two subcategories.

Mental Barriers (Learnability) Voters experience mental challenges when the scheme differs from their previous experience. Depending on the level of differences and the ability of the individual voter to adjust, this issue may be more or less pronounced. Traditional voting schemes use pen and paper which most voters will be familiar with through compulsory schooling. When an electronic voting scheme is introduced, a much larger proportion of voters will not have encountered this type of system before.

Physical Barriers (Feasibility) Traditional voting is based on the ability of voters to read and write on the ballot paper. Voters who cannot read the ballot paper or who cannot write are therefore disenfranchised. The solutions to this can be as simple as providing ballot papers in Braille or an audio interface in the electronic systems. The United Nations Convention on the Rights of Persons with Disabilities, which entered into force in 2008, assures the right to a secret ballot for disabled voters.

Dana Chisnell's invited talk "Counter to Intent: Voter's Mental Models of Alternative Voting Methods"[2] provides one of the best summaries of the importance of useability in a secure scheme. It shows that voters' answers are increasingly distorted in schemes with poor useability, sometimes beyond all recognition. The talk concluded by raising the question *"without usebility, what is being secured?"*

[2]EVT/WOTE 2012

2.2.2 Integrity

In every day English, the term "integrity" has two meanings, the first being a moral quality and the second being "whole and undivided". The cryptographic and information security definition is related to the latter. It means "assuring the accuracy and consistency of data or results". More generally, it can be thought of as protecting the correctness of the system against arbitrary attacks. This further separates into the following two components.

Correctness Voting methods have well defined tallying functions whose various properties can be reasoned. An electronic voting scheme is correct if it computes the function of its underlying voting method.

Integrity against Active Attacks While assuring correctness is a reasonably straightforward software engineering problem, protection from active attacks is far more difficult. Integrity against active attacks should ensure that arbitrary attacks (such as vote substitution, vote tampering, voter impersonation, etc) cannot succeed.

2.2.3 Confidentiality of Votes and Voters

The confidentiality of votes is a complicated issue. The core concept is the protection of the ideals of anonymity and freedom from coercion. The issue is more complicated than just privacy because "If a voter is allowed, but not required, to keep a vote secret, the voter could be coerced by an employer or anyone with power over the voter into casting a certain vote" [BT94]. This leads to the definition of receipt freeness and coercion resistance, properties which are considered necessary for any voting system used in government elections.

Privacy The privacy of a voting system is the cryptographic instantiation that protects and enforces the ideal of anonymity against the system itself (and also against rulers and fellow citizens). The idea is simple: if the system is unable to discern which vote came from which voter, it cannot discriminate effectively. It could still discriminate based on the ballot cast, but this violates the separate ideal of neutrality.

Receipt freeness Receipt freeness was first proposed by [BT94]. They argue that to resist vote selling and coercion, it is necessary that the voter cannot gain a valid receipt of how they voted. Here a receipt refers to information, collected by the voter, which provides strong evidence of how they voted.

A system satisfying this requirement is called receipt free because it is not possible to extract a receipt. It is painstakingly difficult to argue that this property holds, because the voter can take arbitrary decisions to try to gain a receipt which prevents practical modelling. In general, receipt freeness is proven under the assumption of a physical polling booth which prevents any unmodeled leakage or recording of information, but even this assumption is under threat in this era of ubiquitous video recording devices.

Coercion Resistance The definition of coercion resistance varies greatly. It is often used as a synonym of receipt freeness. Some authors, however, use it as the analogue of receipt freeness in the remote setting. It often carries the additional property that the coercer cannot tell if the the voter indeed voted. The property, in the remote setting, is incredibly strong and has only been proven under questionable assumptions. See more details in *Current State of the Art and Problems*.

Other authors, for instance [KT09], prefer definitions which explicitly talk about coercion and counter strategies. A system then has coercion resistance if, for every coercion strategy, there exists a counter strategy for the voter to cast her intended vote in a manner indistinguishable to the coercer from following their instructions. The technicalities of this definition are complicated and many voting systems, even in an ideal model, have very low coercion resistance. There are also serious reasons to doubt that the existence of a counter strategy implies the ability of the voter to follow it.

2.3 Cryptography Related to Voting

Cryptography has existed for thousands of years, with a popular example of early work being the Caesar cipher [Mol06]. These early forms of cryptography are largely synonymous with what is now called encryption. Moving into the modern era, cryptography has become concerned with three major properties:

1. Confidentiality: Keeping information secret and private.

2. Integrity: Preventing changes to information.

3. Authentication: Allowing the source to be corroborated.

In the 1980s two new advances occurred which would facilitate the use of cryptography in voting. The first advance was made by David Chaum in "Untraceable mail, return addresses and digital pseudonyms" [Cha81], who proposed a solution to the traffic analysis problem. The traffic analysis problem occurs when, without knowing what is being said between two parties, analysis of traffic reveals who has been talking to whom and to what extent. The second advance was made by Andrew Yao in "Protocols for Secure Computations" [Yao82] in which secure multiparty computations were introduced. Secure multiparty computation asks the question: Given a series of inputs from multiple sources, can we calculate a function on those inputs without revealing any more information on the inputs than would normally be revealed by the result? This later advance is particularly relevant, as voting is usually defined as a function on the preferences of the voter.

2.3.1 Voting as Secure Multiparty Computation

Cryptographic voting was largely considered as an extension of secure multiparty computation for the next two decades. Much of the research into multiparty computation was easily applicable to voting [Cha81][Cha85][Mer83]. The first cryptographic paper to focus primarily on voting as an application was "A robust and verifiable cryptographically secure election scheme" [CF85a]. This paper was then improved upon in [Coh86] which divided the government into parts to achieve a higher privacy threshold, concurrently by [BY86]; however it allows any single voter to disrupt the election. Chaum brought together "sender untraceability" [Cha88a] and "blind signatures" [Cha82] in [Cha88b] to create a verifiable voting scheme. At the same time, various efforts in multi-key cryptosystems were being applied to voting [Boy90].

A major transformation came to electronic voting in 1994 when Benaloh et al presented "Receipt-Free Secret-Ballot Elections" [BT94]. The paper argued that being able to prove your own vote allows vote selling and coercion—hence privacy, as it is normally defined, alone is not sufficient for voting. "With voting booths, a voter can make promises to employers, accept bribes, and belong to organizations which are committed to a particular vote. Yet the voter,

in the privacy of a voting booth, may cast the opposite vote without fear of repercussions or recrimination." To reflect this, the new term Receipt-Freeness was introduced, meaning the voter should not be given a receipt that allows them to prove how they voted. It is largely this property, along with the uncontrolled environmental issues, that led electronic voting back to the polling booth.

One of the primary issues with the schemes proposed in this era was the inheritance—and in some cases uncritical acceptance—of the basic cryptographic assumption that users have a way to do complex mathematical calculations securely. This is not true in practice, and the deviations between the security models and reality open the schemes to a range of attacks.

Many of the primitives and protocols which were invented at this time are still used as components, but are in and of themselves too abstract for use in voting. A good example of this is Mixnets [Wik04]. Despite the significant progress made in this era, Cryptographic voting as multiparty computation failed to achieve the verifiability and uncoercibility that was and is desired.

2.3.2 Voting Back in the Polling Booth

While several schemes were quickly proposed that obtained the receipt freeness property[3], for instance [BT94][HS00a][CGS97], they achieved this in ways that resulted in less than ideal useability. The issue of verifiability and direct coercion at the time of voting was first addressed practically by David Chaum in "Secret-Ballot Receipts: True Voter-Verifiable Elections" [Cha04]. In this new style of scheme, the voter's computer is not considered synonymous with the voter, and the human can verify the correctness. This presented a paradigm shift away from treating voting in much the same way as any other multiparty computation. Evidence for the abruptness of this shift can be seen in that Chaum makes minimal reference to earlier voting schemes in his paper.

Prêt à voter In "Secret-Ballot Receipts: True Voter-Verifiable Elections.", Chaum made use of visual cryptography to implement a cut and choose protocol [Cha04]. However, the visual cryptography used was complicated and unwieldy from the perspective of most voters. Peter

[3]Some of these schemes initially contained errors which have since been rectified.

Ryan's insight in "A variant of the Chaum voter-verifiable scheme" [Rya05] was that the separation of the data into two sets—that provided no information without the other— allowed the cut and choose protocol to work, without the visual cryptography. He used this to construct a scheme based on cyclic shift (or full permutation) that allows the same idea while appearing nearly indistinguishable to existing paper voting. This idea was quickly expanded upon in "A Practical Voter-Verifiable Election Scheme" [CRS05]. Ryan's new variant was called Prêt à Voter.

Prêt à voter continued to see development in a wide variety of directions, which we will discuss in more detail in 2.5. A particular landmark was the choice to use it in the Victorian State elections [BCH+12a][BCH+12b]; this was trialed in the 2014 elections [RST15][CRST15a].

In addition to Prêt à voter there was a collection of schemes that were important at the time, and in the development of the field, but which are not covered here because they are they are no longer relevant (at least to our area of research). Examples include "A practical voting scheme with receipts" [KKLZ05] and "Practical high certainty intent verification for encrypted votes" [Nef04].

Kleptographic attacks Many of these schemes (including Prêt à voter) were readjusted because of a series of data exfiltration attacks presented in "Kleptographic attacks on E-Voting schemes" [GKK+06]. This was a Kleptographic attack [YY96] that exploited the randomness utilised in cryptographic primitives employed in voting schemes. It effectively allowed corrupted devices to securely transmit information through the system without detection.

Interestingly, kleptographic attacks have become a popular area of cryptographic research in recent years, as a result of the Edward Snowden revelations[4]. These revelations of widescale exfiltration and mass surveillance have motivated significant general research into solving this problem [BPR14][DGG+15][BH15][AMV15].

2.3.3 Human Cryptography

The properties discussed earlier (privacy, integrity) should hold even in the face of a range of adversaries, including the computational device assisting. This presents a problem to traditional

[4]Much has been written on the revelations, one collection of the revelations can be found at https://edwardsnowden.com/revelations/.

cryptography which assumes that the user has access to secure computational devices. The solution to this is human cryptography which aims to achieve secure systems which do not rely upon the devices used.

We use human-computable cryptography here to refer to any human involvement in a protocol which achieves a cryptographic goal. Hence, a cryptographic scheme is considered human-computable cryptography if it can be computed, or at least verified, without the aid of a computational device. Early examples include the dining cryptographers protocol [Cha85]. We look at two modern examples with possible applicability to electronic voting.

One interesting area looks to use both humans and machines in cryptographic protocols. An example of this is "Security Ceremonies (Including Humans in Cryptographic Protocols)" [Rad13] which allows the human to verify the freshness of the protocol run. An application of this to voting which does not require a second channel of communication, like in Remotegrity [ZCC+13], would be valuable.

Another example of what we would consider human cryptography is Rivest's ThreeBallot scheme [Riv06]. Rivest himself preferred the term secure voting without cryptography. Unfortunately the scheme fails to provide receipt freeness. ThreeBallot is an example of human-only encryption. A recent paper called "Cryptographic protocols with everyday objects" [HST14] is another good example of the parts of these fields that might be relevant to voting.

Recently Boyen [Boy16a][Boy16b] presented the first human-computable public-key cryptosystem. The system requires a computer to encrypt but can be decrypted by an unaided human. This has the potential for significant impact on electronic voting, but its precise applicability is currently unstudied.

2.3.4 Why Cryptography?

Depending on their background, some readers may assume that cryptography is all about creating secure channels for internet banking. While the secure channels are useful for electronic voting, they are only the beginning of cryptography's usefulness.

We aim here to provide an intuition for why cryptography is useful in achieving many of the natural properties of voting. In particular, in this section we focus on a less known area of cryptography called zero-knowledge proofs and their applications to voting. Zero-knowledge

proofs were introduced in "The Knowledge Complexity of Interactive Proof-Systems" (sic) [GMR85]. They allow statements to be proved without providing additional information. They usually contain an interactive component which makes them nontransferable. For voting this facilitates a series of benefits:

1. Voters can prove that their ballot is correctly formed without proving which ballot they are casting.
2. Voters can be given a receipt to ensure their ballot is cast and processed correctly which cannot be used to prove to others how they voted.
3. Authorities can prove that ballots are mixed correctly (no ballots are modified, removed or added) without telling how they are mixed.
4. Authorities can prove that they correctly decrypted the ballots without releasing the key.

2.4 Cryptographic Voting Schemes

2.4.1 Overview

Cryptographic voting schemes are often categorised into three types: (1) those using mixnets [Cha81] [PIK93] [SK95] [Abe99] [CRS05]; (2) those using homomorphic encryption [CF85b] [Ben87] [CFSY96] [HS00b]; and (3) those using blind signatures [Cha88c] [FOO93] [Oka96] [Oka98a].

A mixnet is a cryptographic system that accepts input, normally in batches, and produces an output containing the cryptographically transformed, permuted input. The transformation and permutation by the mixnet prevents the tracing of input to output. For a survey on the topic see [SP06]. In the context of voting it prevents the tracing of decrypted votes to the submitting voter. Homomorphic encryption allows computations to be computed on the ciphertext. This allows the votes to be tallied securely and then the result alone decrypted. Blind signatures allow the signing of the blinded votes by the authority. The votes can then be unblinded and submitted anonymously.

- Mixnet-based schemes in general allow arbitrarily expressive voting at a relatively fixed

cost, since tallying is done on the mixed votes in plaintext; however there is a delay in tallying due to the necessity of applying verifiable mixers after the election. These schemes also have lower privacy than homomorphic schemes since the set of votes is released rather than just the result of a function of those votes.[5]

- Homomorphic schemes facilitate a higher possible level of privacy since individual votes are not revealed, only the final tally with suitable proofs, but tallying in this approach is an even more expensive proposition in most cases. Homomorphic schemes are generally only practical in first past the post or other unexpressive voting systems.

- Blind signature based schemes are able to use less expensive implementations of anonymous channels than mix-net based schemes, hence they are usually more efficient. Ring signature based schemes also fall into this category but do not require a central authority.

2.4.2 Beginnings

Applications of cryptography to voting originate with Chaum [Cha81] in 1981, who later combined notions of "sender untraceability" [Cha88a] with blind signatures [Cha82] to construct an effective cryptographic voting scheme [Cha88c]. The early literature flowing from those papers is mostly concerned with the use of cryptographic primitives, such as blind signatures and anonymous channels, to construct voting schemes—of which some examples are particularly well known [Cha88c] [FOO93] [Oka96] [Oka98a].

2.4.3 Receipt Freeness

A rift arises from the emergence of "receipt freeness" advocated in [BT94], which is the requirement that no transferable receipt or proof of vote be made available to the voter: a condition that is essentially incompatible with remote voting[6]. In [Oka98a], receipt freeness is conflated with the notion that the the voter is trying to avoid coercion, which is an objective better fulfilled by deniable encryption [CDNO97] than receipt freeness *per se*. From that point forward the literature splits, with some schemes—of which Civitas [CCM08a] is a good

[5] Mixnet schemes' privacy is similar to many paper based systems if one assumes an idealised ballot box.

[6] Supposed counter examples include JCJ/Civtas. However to realise the assumptions of such schemes in-polling booth interactions seem to be required. Hence, the instantiations of such schemes are better thought of as hybrid, rather than truly remote.

example—aiming to provide coercion resistance by requiring trusted devices and/or complicated and often infeasible user interaction. The other offshoot of literature aims at providing either a smooth user experience or enabling voter verification without trusting computational devices for integrity [Adi08] [ZCC+13].

2.4.4 Supervised vs Unsupervised: A Major Divide

There is a significant difference between electronic voting systems which occur in supervised environments, such as polling booths, and those that occur in unsupervised environments, such as at home or other remote locations.

Supervised electronic voting systems have been and are used to elect governments. These systems achieve properties very similar to paper-based alternatives and have the potential to perform significantly better than paper. In particular the issues of voter eligibility and compulsory privacy in supervised electronic voting is equivalent to paper alternatives.

Unsupervised electronic voting systems, on the other hand, do not provide the standard protections for voter eligibility and compulsory privacy. This opens the possibility for vote selling and coercion to become significant attacks. Schemes that claim coercion resistance often have other undesirable properties which they claim are justified due to the value of coercion resistance. If in fact these schemes do not satisfy coercion resistance, they are considerably worse than other alternatives and should not be used in practice.

The schemes that "prove" coercion resistance do so without capturing the idea that the voter may want to prove their vote, in which case all bets are off. It is not sufficient to define a model and process, and show that nothing in that model using that process breaks coercion resistance. A willing voter has many options to record the voting process—outside of its model—in an arbitrarily convincing way. Furthermore, the zero-knowledge proofs that are often used for credential distribution do not enforce non-transferability in the remote voting setting.

JCJ/Civitas The process of remote voter registration fundamentally fails in JCJ/Civitas to satisfy the robust definitions of receipt freeness and coercion resistance. In particular, JCJ/Civitas relies on a number of clearly false but necessary assumptions; we detail what happens when two of these are instantiated in the real world.

We will now describe Civitas in simplified terms. Civitas involves five kinds of agents: a

supervisor, a registrar, voters, registration tellers, and tabulation tellers. It uses an underlying log service and digital signatures to ensure the security of the data used for verification. It also uses a trusted device to handle the mathematically complicated cryptography. After an initial setup, the registration tellers send the voter's private credential through the internet to the voter in a process that utilises the voter's trusted device. The voter can run a local algorithm to generate fake credentials which are indistinguishable to an adversary; the voter hands over a fake credential whenever they are coerced. The voter submits a vote using their real credential and finally the votes are tabulated.

"Trust Assumption 1" "An adversary cannot simulate a voter during registration" is unjustified in the remote setting; one might as well assume that the adversary cannot simulate the voter at the time of voting.[7]

"Trust Assumption 2a" "Each voter trusts at least one teller," is a direct contradiction with authority independence. We have already, and will continue to, argue that this is false for many real democracies, which tend to be dominated by a small number of powerful and colluding parties.

"Trust Assumption 2b" "and the channel from the voter to the voter's trusted registration teller is untappable." The existence of an untappable channel in the remote environment is a currently, and possible fundamentally, unsatisfiable requirement. Satisfying this requirement is therefore fundamentally at odds with the scheme's claim to be usable in a remote setting.

Attack 1 The coercer attends the voter right at the start of the election period and orders them to contact the tellers for their credentials then and there. The voter cannot signal the presence of the coercer to tellers without detection unless it already has a secret code, which becomes a circular argument. Since the credential is then displayed along with proofs the coercer learns the code. This attacks breaks trust assumption 2b that at least one channel is uncontrolled by the adversary. Since all channels include the same final

[7] There is a time escrow between registration and voting which might justify this, but would reduce Civitas from an election system to a time expander for an already secure election system.

display element (the voter's computer monitor) this assumption effectively resolves to a perfectly secure display, among other things.

Attack 2 The coercer removes the trusted device from the voter after registration but before voting. This breaks both trust assumption 1 and 2, however it is a perfectly valid way for the voter to transfer their vote.

Attack 3 The assumption of a cryptographically secure untappable channel has no known instantiation. The various types of deniable encryption-based systems either contain ephemeral data which must be destroyed, or would fail if certain adversarial provided ephemeral data was used. The attack here works by exploiting the failure of the channel used to provide the deniable properties required from an untappable channel.

We think that coercion resistance or a weaker property may yet be possible in remote voting, but the primitives required do not appear to exist.[8] While many "coercion resistant" remote voting schemes are of significant interest and provide valuable theoretical results, the form of abstract analysis which many schemes present fails to explain or justify the often unobtainable physical assumptions they are implicitly making.

2.5 Current State of the Art

Here we aim to summarise the current state of the art in electronic voting. Once again we are unavoidably constrained to limit our scope. Therefore, we focus on those schemes which have been used in government-binding elections, or are being planned for such use; as we consider these to be of the most practical interest. We further focus on those schemes which achieve strong security properties, since we consider these to be of the most theoretical interest.

2.5.1 Prominent Current Schemes

Helios Helios is a remote verifiable voting scheme introduced by Ben Adida [Adi08] in 2008. As well as the published specification, the work included an open source code repository and

[8] Even if humans could compute public key cryptography in their heads, without leaking side-channel information, this does not appear to be sufficient.

a website that hosted elections using the system at no cost. It seems likely that this easily accessible working example provided much of the popularity that Helios has and continues to enjoy.

An interesting extension of Helios is Zeus [TPLT13], which was used in a nationwide election of university officials in Greece. This system was designed to be secure against fierce opposition from a significant minority of voters. It did this by granting additional powers to the election authority, partially trading off integrity and privacy guarantees for additional robustness.

Unfortunately Helios provides no protection against coercion. Voters can both prove their vote and be observed or controlled by a coercer at the time of voting. Helios makes this clear to voters by including a button labelled 'coerce me'. This button is removed in many deployed variants but the ability to easily coerce and vote sell remains.

Scantegrity Scantegrity [CCC+08] was first proposed as an extension that adds verifiability to optical scan systems. However, the variant that has had the largest impact on electronic voting is Scantegrity II, which uses invisible ink to hide confirmation codes. This variant has been used in binding government elections in Takoma Park, Maryland, USA [CCaJC+10].

An extension of Scantegrity II in the remote setting is Remotegrity [ZCC+13], which uses standard snail mail to send a sheet of confirmation codes to the voter. The voter can then use their computer to send these codes back to the server in such a way that their computer cannot alter or understand what is being sent with non-negligible probability, and the voter can prove that the election authority cheated if it does cheat.

STAR-Vote Another recent example of voting schemes for real world application is STAR-Vote [BBK+12]. The name is a shortening of "A Secure, Transparent, Auditable and Reliable Voting System". It is the result of a large group of researchers in the field who were contacted by Travis county in Texas. It is a dual system, paper and electronic, which is designed to allow a fallback in case of attack.

Prêt à Voter Prêt à Voter was introduced by Ryan [Rya05] based on Chaum's "Visual Cryptography" [Cha04]. The key innovation, which is at the heart of Prêt à Voter, is to vary the

candidate order. By doing this, Prêt à Voter provides privacy equivalent to the cryptosystem—used to encrypt the candidate order—unless the trusted devices are corrupt. We refer the reader to the overview in 3.2 for a fuller introduction.

2.6 Open Problem(s)

In discussing open problems we will first discuss the adversarial powers that need to be considered. We then discuss what combination of these powers the schemes above are secure against. We argue that no voting scheme can be made meaningfully independent of the voters. This leaves authority—and device—independence to be considered, and that is the problem we choose.

2.6.1 Threat Model

When defining a security model in cryptography both goals and powers must be considered. We have already discussed the goals in the requirements of voting. This leaves powers of the adversaries to be considered. From a security point of view, voting presents an incredibly hostile environment. Suggested analogies include securing a plane where the pilot is trying to crash it and where crashes are hard to detect, and operating a bank which is not allowed to know the amount of money in its clients' accounts and the clients are not allowed to prove how much money they have[9]. We consider below three main groups of adversaries beyond the general external adversary.

Authority In elections the authority is often chosen or controlled by the current governing body. If the election aims to choose the next governing body, this is a clear conflict of interest. There is the additional issue that any party with high stake in a particular election result is tempted to compromise the authority. It is common to divide many of the functions across a set of parties which are believed to have competing interests. In practice, while it may be possible to layer enough protections around the authority, it is far more convincing and verifiable to remove as many trusted functions from this party as possible.

[9]For more detail on these and other analogies see the introduction of Ben Adida's thesis [Adi06].

Devices Most cryptographic protocols do not make the distinction between users and the devices which execute the mathematically complicated cryptographic primitives. Given that the devices are often under the control of a single party or provided by non-technical users, there is a reasonable likelihood that at least some are malicious. Voting systems should, therefore, be secure even if all or most of the devices involved are malicious.

Voters There are two primary reasons not to give voters a proof of how they voted. First, research into human decision making has shown that in large elections people can be easily convinced to vote against their own best interest through vote buying [KK98]. Secondly, any coerced voter will likely act against their own best interests in order to avoid negative consequences. If no proof of which vote was cast is available, then the adversary will be unable to tell between a cooperating and un-cooperating voter, which prevents meaningful implementation of these attacks.

2.6.2 Authority Independence

Complete authority independence is normally not obtained in voting schemes; rather, the authority is divided among several parties which would have to collude to attack the system.[10] Authority independence in terms of privacy can be obtained in one of two ways. First, the authority can learn votes but not voters. Secondly, the authority can learn voters but not votes. The primitive of a mix-net is designed to allow for secure transformation between these states. However, since it relies on its own set of authorities, it transfers rather than solves the problem.

2.6.3 Device Independence

Device independence holds if the device(s) assisting the voter can perform no other attack than denial of service. They should not be able to attack integrity or privacy. At the moment, the only viable solution to this problem appears to be various variants of code voting. Remotegrity [ZCC+13] is perhaps the most prominent example in this class of schemes. However, code voting is a symmetric primitive which fundamentally requires the authority to be able to identify and decode the ballots. This enables the authority to break privacy, which is a violation of authority independence.

[10] When the set of authorities includes the complete set of voters, then perhaps, complete authority independence could be said to be achieved.

This is related to the field of human identification protocols. However, human identification protocols are significantly weaker in their adversarial model. Still, it is possible that ongoing developments in that field might have a profound impact on electronic voting.

2.6.4 Voter Independence - Coercion Resistance

Coercion resistance is achieved if the system ensures confidentiality regardless of the actions of the voter. Another name for this property is voter independence.

2.7 Conclusion

The requirements of voting are complex and securing electronic systems is hard. Further work is needed to improve electronic voting systems and allow them to adequately meet the requirements of higher stake elections, particularly government elections.

Modern end-to-end verifiable electronic voting systems have matured to the extent that undetectable attacks on integrity are no longer a prevalent issue. The issue of non-receipt freeness has also been largely solved (though modern cameras are introducing new side channel attacks). However, no modern system provides privacy in regards to the election authority; that is to say, in each system there is at least one party/device (normally the one printing ballot papers) who learns how voters vote. While this problem can be overcome—see research progress in Chapter 3—the inability to obtain these two properties simultaneously is a significant gap in the literature.

Authority independence in low-stakes elections is a significant issue. Prominent schemes like Helios rely upon trusted mixers—set up specifically for the scheme—to protect privacy. There is a need to investigate ways to achieve these properties in ways that better distribute trust and increase practicality. There is also a need to investigate new cryptographic primitives to support electronic voting schemes.

Device and Authority Independence are significant and related unsolved issues in remote and in-polling booth voting. In the remainder of this thesis, we will present various completed research projects that address these issues.

Chapter 3

Truly Multi-Authority *'Prêt-à-Voter'*

Never trust a computer you can't throw out a window.

(Steve Wozniak)

3.1 Introduction

Electronic voting schemes are being implemented in government-binding elections to enable fast tallying and reduce costs. Advocates of electronic voting for government elections argue that it will increase efficiency, increase participation and turnout, reduce queues, and speed up counting. Significant segments of the academic community, however, are concerned with the security of electronic voting. They are particularly worried about the possibility of massive, low cost, and undetectable attacks. Online electronic voting schemes depart from the controlled environment of the polling booth. This opens many security threats around vote selling and coercion, as well as issues of voter identification. For this reason, many argue that electronic voting—if it must be done at all—should remain in the polling booth. They advocate, furthermore, for systems with end-to-end verification of the election result. These systems provide strong evidence for the correctness of the election result, ideally without either false positives or false negatives.

However, even these end-to-end verifiable schemes are not without issues. While the integrity is often well protected, privacy is much harder to guarantee. One of the most significant issues with these schemes is how to print or display the ballot paper without jeopardising privacy. In several of these schemes, freshly generated unmarked ballot papers contain critical information which, combined with public "bulletin board" information, breaks ballot secrecy.

Three of the most prominent end-to-end verifiable electronic voting schemes are Prêt à Voter, Scantegrity II and STAR-Vote. These schemes all rely on a trusted authority to act honestly and perfectly, and to ensure privacy. The exact way in which this occurs is different in each of the above schemes. In STAR-Vote the authority provides a Direct Recording Machine (DRM) which learns the ballot directly. In Prêt à Voter, the voter is required to receive some secret and unique information (a permutation of candidates) in order to fill and cast their ballot. Similarly, in Scantegrity II [CCaJC$^+$10], the voter must receive secret confirmation codes. The requirement that this information must be kept secret creates difficulties in the generation and transportation of ballot papers. The tamper-evident ballot papers of Scantegrity II should provide strong evidence to the voter that the information has been transported securely but provides no guarantees about privacy against the printing authority. There has been work on secure printing [ECHA09a], nevertheless in that instance the voter receiving the ballot paper is unable to readily verify the privacy of their ballot paper.

We present a practical solution which uses re-encryption inside the polling booth to print ballot papers in a privacy-preserving manner. This makes practical, at a user rather than computer level, multi-authority voting. We then apply this solution to *Prêt à Voter*, a state-of-the-art electronic voting system trialled in a recent Victorian state election. We propose two approaches: one with higher security and another with stricter useability constraints. Our variants achieve threshold device privacy without relying on prior secrets. The primary benefit is that ballot papers no longer pose a privacy risk. The solution has the major benefit of resolving the conflict between auditability and forward secrecy of printers, a problem left open by the most recent work in this area.

Additional benefits include practical privacy from compromised polling-place devices, while preserving receipt freeness against a more general adversary. Although we do not provide privacy against a wholly compromised authority, a voter needs honesty from only one of the machines at the polling site for secrecy.

3.1.1 Background

The central issue which dominates the security of voting is how (simultaneously) to achieve integrity and privacy. Any solution to this issue seems to require that the voter obtain some information which the adversary does not. Since any mechanism that allows a voter to verify

their vote will also convince a coercer if they both have the same information, the voter needs to receive some information that the adversary cannot. In cryptographic terms this means the voter must have some channel to the election authority which is confidential from the adversary. A standard secure channel is confidential and authenticated. An untappable channel is similar to a secure channel but its privacy and integrity are unconditional and compulsory. An untappable channel—in at least one direction—between voter and authority seems necessary for receipt freeness [HS00b]. Polling booths are often kept in schemes designed for government-binding elections to realise this constraint.

The issue of privacy against the adversary is further complicated when we consider the election authority itself as an adversary. To mitigate and control this issue, the role of the election authority is often divided among a collection of parties whose interests are in conflict. The preferred mechanism for this is threshold cryptography. However, this does not defend privacy from the machine that encrypts the vote (as in Wombat [Gru12], STAR-Vote [BBK+12], or the Moran-Naor scheme [MN10]) or prints the ballot paper (as in Prêt à Voter and Scantegrity II). In fact, threshold cryptography further complicates the voting process by requiring an additional trusted computational device, in the absence of human-computable threshold schemes.

Dividing "trust" amongst multiple entities creates a strong difference between privacy against the election tellers (those holding the threshold key parts) and privacy against the poll-site machines or printers. We show how to defend against a compromise of all but one of the machines that a voter uses in a polling place. This does not protect against a completely corrupt election authority who sets up the polling booth and hence controls all of its computational device(s). It does, however, protect against *ad hoc* compromises of individual poll-site devices.

Three of the prominent in-polling booth voting systems are Prêt à Voter [Rya05], Scantegrity II [CCC+08] and STAR-Vote [BBK+12]. Each represents a different approach to in-polling booth computer-assisted voting. Each of these approaches has a largely disjoint set of possible solutions to achieving privacy against corrupt devices. We review these briefly.

STAR-Vote uses a device to encrypt votes directly from human input. Such a device necessarily learns the votes, so any solution attempting to achieve privacy against corrupt devices in STAR-Vote would seem to require the use of multiple devices to encrypt votes.

Scantegrity II relies on optical scan systems and provides end-to-end verifiability of election results. It does this through printing confirmation codes on the ballot paper which the

voter uncovers as a part of voting. These confirmation codes later appear on the Bulletin Board allowing voters to confirm their vote. In Scantegrity II, the use of static (non-randomised) confirmation codes prevents re-encryption. Since re-encryption is not possible, cast ballot papers cannot be further anonymised through mixing.

Prêt à Voter uses hybrid human-computer cryptography to achieve a high level of practicality and privacy. The issue of privacy against corrupted devices in Prêt à Voter exists primarily in the way ballot papers are generated. Ballot papers cannot be generated directly due to the risks of privacy breaches and kleptographic [GKK+06] attacks. The kleptographic problem is generally resolved by distributing ballot paper generation information across a set of tellers, as in [CHJ+13]. However, solutions of this sort still use a single physical printer which must be trusted for privacy.

So, while it is possible to divide the authority among election tellers and to suggest constructions of secure channels, all of these solutions currently require some single device (printer or ballot marker) to be trusted. That device presents sufficient information to the voter to enable them to vote, and in doing so, that device learns sufficient information to recover the voter's selection (at least once verification information appears on the Bulletin Board). A solution was proposed in [ECHA09a] to make use of visual cryptography to allow multiple printers to construct ballot papers. Another approach involving the use of multiple re-encryption clerks was suggested in [Rya10], but this was later shown to be broken in [RT09] because the large permutations leaked are likely to be unique. While Ryan and Teague in the same paper [RT09] presented a fix, their solution is of partial applicability and the current literature around implementing voting schemes has not incorporated it.

3.1.2 Our Solution

The core idea behind our solution is to allow optional re-encryption on already ***separated*** ballot papers to provide privacy against corrupted devices. In a nutshell, a Prêt-à-Voter separated ballot paper is the one half of the paper ballot that is about to be cast; see Section 3.2 for details. Our variants are similar to the theoretical voting system of Hirt and Sako [HS00b], however, Hirt and Sako do not make the distinction between the voter and their computational device.

It is precisely the challenge of practically unraveling how to achieve security to the voter,

without trusting the device for privacy, which is our primary contribution. In both of our variants we construct an anonymous channel using a set of tellers, much like a mixnet. However, in contrast to mixnets, the re-encryption occurs on individual ballot papers, and is driven by voter action. The scheme has some features in common with the trusted (re)randomisers of Lee *et al.* [LBD+03] and Aditya *et al.* [ALBD04], but we use no trusted components or authorities for the randomisation, and we preserve cast-as-intended verification.

The most important property we aim to achieve with our system(s) is to require ***no prior secrets***. It aims to capture an adversary's power to control all items and data that the voter brings into the polling booth.

Principle 1. ***No prior secrets****: The adversary has full knowledge and control over all the information handed to the voter before entering the polling booth.*

We note that other end-to-end schemes such as STAR-Vote, and Prêt à Voter as implemented in Victoria (Australia), do not require prior secrets to be passed to the voter. Instead, they contain single devices which are only procedurally prevented from breaking privacy. Conversely, we define ***full threshold device privacy*** to reflect the property of a scheme which maintains privacy against single-device attacks.

Principle 2. ***Full threshold device privacy****: A voting system has full threshold device privacy if, provided that at least one device is honest, the voter's privacy is assured.*

We also add the additional constraint that the devices must not be networked. This allows procedural measures against kleptographic and other attacks targeting and originating from the devices to be more readily implemented. This may appear to be a slightly odd definition; however, particularly in the second variant, the devices should be as minimalist as possible. Schemes satisfying this definition can more readily realise minimalist devices. We provide further guidelines on devices (EBMs) in Section 3.6.1.

Principle 3. ***No networks****: No device inside the polling booth requires network access to any other device, local or remote, to function during the election.*

Receipts in end-to-end verifiable voting schemes are given to voters to allow them to check that their votes were counted. This differs from the term "receipt", as used in receipt freeness, which allows the voter to prove how they voted. We make the assumption that the receipts,

here using the first sense, that the voters receive, are publicly known and linked to them. Since voters are encouraged in most proposals to share their receipts with as many interested parties as possible, this assumption seems reasonable.

Assumption 1. ***Public-Receipts****: The information provided on the Bulletin Board and on receipts to enable vote verification, is publicly available, and the links to voters are known.*

Our first variant relies upon a human mental calculation assumption very similar to that of Prêt à Voter. In standard Prêt à Voter, it is assumed that a voter given a permutation of candidates can apply the mental permutation of their preferences and create a ranking. This is a reasonable assumption[1] and is slightly modified for our variant; we assume the voter can take two listed permutations and compose them.

Assumption 2. ***Mental calculation****: Voters can compose two permutations.*[2]

We also define a second variant that does not rely on the mental calculation assumption, making it more suited for the more complicated Single Transferable Vote or Instant-Runoff Voting contests. To achieve ***threshold device privacy*** and ***no prior secrets*** without using voter ***mental calculations***, we require a slightly stronger polling booth assumption which we call ***device anonymous polling booth***. This assumption means that the devices have no means to identify the voter other than from the information that passes between them. The full implications of this assumption should not be underestimated. As Benaloh notes in [Ben13], this kind of assumption is to some degree untenable despite its widespread use.

Assumption 3. ***Device anonymous polling booth****: A voter interacting with devices in a polling booth does so over an anonymous untappable channel.*

No new information is revealed to general adversaries since the ***separated*** ballot papers, now to become re-encrypted and mixed, were public information in the original Prêt à Voter. A completely corrupt authority, or an external attacker, may still compromise privacy, but only if all the machines that the voter uses in the polling place are corrupt. We stress that since in-booth mixing is optional, it cannot guarantee receipt freeness against the election authority. There is a

[1] It is reasonable for small permutations. Nevertheless, the Victorian Prêt-à-Voter variant used a machine to assist voters with this task because computing a permutation of its 50 candidates was deemed too difficult.

[2] The composition of two permutations resulting in their product is a basic algebraic operation. This operation is a special case of the pointwise sequential evaluation of two enumerated functions.

straightforward solution to achieve this last property but it comes at significant useability cost. The straightforward solution is to make the checking interactive as is normally done in End-To-End (E2E) verifiable electronic voting. The voter can either proceed with the ballot paper or audit but never both. Since the audit is now interactive, the voter does not gain a proof of their actual vote, which could otherwise be used as a receipt.

3.1.3 Motivation and Contribution

The in-polling booth voting schemes we have discussed are an important area of study because they are already being used in government-binding elections and are planned for expanded use.

The trust assumptions on tellers with a component of the threshold decryption key—which allows them to collude to break the privacy of any voter—is normally based on the assumption that the tellers are drawn from all political parties and hence have no mutual interest on which they might collude. This assumption—while questionable—is far superior to trust in a single device.

Motivation

The bottleneck of the single device able to break privacy in the polling booth is of great concern and largely unaddressed. We propose to allow concerned voters to interact with the set of tellers inside the polling booth, thus removing their reliance on a single device. Our scheme provides no privacy against a fully corrupt set of tellers; it does however provide privacy against a partial compromise of the poll-site devices.

None of these current voting schemes being implemented in government-binding elections—[CCaJC$^+$10] [BCH$^+$12b][BBK$^+$12]—offers privacy against a compromised printing or voting device. This is because in each of the current schemes there exists some device or set of devices—provided by the election authority, not by the set of tellers—that learns votes. While this may be acceptable in some situations, there is certainly a need for solutions that do not require this trust assumption.

Contribution

We make practical, at a user rather than computer level, multi-authority voting. We do this through a process of re-encryption by mixers which act on individual paper ballots, possibly after marking, but before scanning, to prevent information learned in printing from being readily used to reveal votes. This is our primary contribution.

This approach has the following advantages over previous work:

1. All devices with which the voters interact must collude to break privacy (barring an active voter coercion attack by the authority itself).

2. Freshly generated ballot papers reveal no useable information—provided at least one of the re-encryption mixers is honest—about how the vote will be decrypted; therefore they can be printed, transported and distributed without additional complication.

3. Significantly reduced audit overhead compared to previous schemes with this level of privacy, since audited ballot papers need not be discarded. The audit overhead is still higher than standard Prêt à Voter, which is the price we pay for a higher level of privacy.

4. No scheme which would have otherwise achieved receipt freeness or coercion resistance will lose those properties through the use of our extension. This follows since all information the in-booth mixers see used to be public in the original scheme.

5. The level of re-encryption can be altered by the voter according to their personal privacy desires without allowing vote selling.

6. In the context of the Victorian election system vVote [CRST15b], our approach would mitigate the lack of printer forward secrecy because re-encrypted receipts keep the votes secret even if the printer's data is later exposed.

The following additional benefits are also gained, if the ballot papers are generated in the polling booth:

1. Since the ballot paper, as handed to the voter, contains only a unique serial/seed (preventing kleptographic attacks) it can be constructed into a valid vote for any race of the

election. Since ballot papers can be used in any race in the election, the issue of ballot distribution[3] is eliminated.

2. The cryptographic cost of generating ballot papers can be distributed.

Practically, our improvements would directly provide higher levels of privacy, e.g., to the version of Prêt à Voter recently used in a parliamentary election in the Australian state of Victoria [CRST15b]. The white papers by the vVote team have always described a networked printer or electronic ballot marker with access to enough information to break ballot secrecy. Our solution allows a voter to interact with re-encryption mixers and remove this trust in the privacy of those devices, as long as not all of them collude.

Limitations If the election authority can observe the voter in the polling booth, there is no privacy, however this is true of all schemes discussed. Indeed, even in the remote setting, JCJ [JCJ05] and Civitas [CCM08b], among others, assume that the voter cannot be observed at key points of the process. The more interesting and controversial assumption we make is that the voter will not subject themself to coercion and provide the proofs of correct mixing to the election authority, although he or she would be able to do so. In other words the scheme does not have receipt freeness against the election authority, but it does against everybody else.

It would be a simple design decision to make the proofs of correct mixing interactive and hence close this issue; however for any significant number of mixers this would quickly become impractical for voters. We therefore limit privacy to the weaker definition, in order to maintain a higher level of practicality. The result of these limitations is a split in the level of coercion resistance achieved. In regards to a general adversary, the scheme achieves the same receipt freeness as standard Prêt à Voter. However, against corrupt devices it achieves the weaker property of privacy, which Prêt à Voter did not provide.

3.1.4 Outline

Having introduced the problem of ballot paper generation and printing in the context of the three most prominent verifiable in-polling booth electronic voting schemes; we went on to motivate

[3]The issue of ballot distribution occurs when insufficient ballots for a particular race are present in a given polling place. This issue occurs most frequently in elections where voters can choose which polling place to attend, as is typically the case in Australia.

the importance of the problem. The definitions and assumptions under which our solutions work were then discussed above. The advantages and limitations of our contribution have just been discussed.

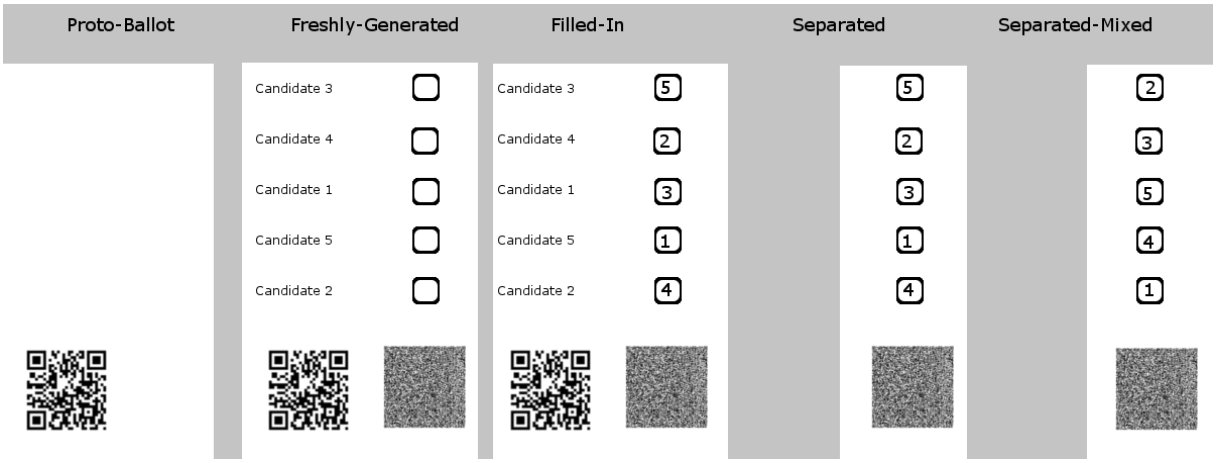
For those not familiar with Prêt à Voter, section 3.2 provides a brief overview, including problems that can occur. Section 3.3 provides a brief overview of our method and how it is used in our two variants. Section 3.4 introduces Variant 1. Section 3.5 introduces Variant 2. Section 3.6 discusses practical matters including the issue in auditing the variants. Section 3.7 concludes this chapter and discusses future work.

3.2 Prêt à Voter

Before entering into the details of our variant we provide an overview of Prêt à Voter, with particular emphasis on the elements that allow re-encryption.

Prêt à Voter was introduced by Ryan [Rya05] based on Chaum's "Visual Cryptography" [Cha04]. The key innovation, which is at the heart of Prêt à Voter, is to vary the candidate order. By doing this, Prêt à Voter provides privacy equivalent to the cryptosystem—used to encrypt the candidate order—unless the trusted devices are corrupt.

Figure 3.1: Summary of Prêt à Voter ballot paper states.



A Prêt à Voter ballot paper, ready to be filled in, is shown in figure 3.1, under the description ***freshly generated***. It consists of a left hand side (LHS) and a right hand side (RHS). The LHS contains the list of candidates in a certain ranking in both human and computer readable forms. The RHS contains boxes in which the selection (or ranking) can be marked and an encryption

of the candidate order under a threshold-cryptographic key of the election tellers (encoded as a high-density QR code).

"Conventional" Prêt à Voter ballot papers can be thought of as existing in three distinct states: ***freshly generated***, ***filled-in*** and ***separated***. For Prêt à Voter to be receipt free, it is clear that the ***separated*** ballots should not reveal the votes. In addition, it is clear that ***filled-in*** ballots will reveal votes. ***Freshly generated*** ballots are interesting, since they do not reveal votes on their own; however, they contain the relationship between the permutation and the ciphertext/serial. Since this serial/ciphertext will later appear on the Bulletin Board next to a ranking, ***freshly generated*** ballots *do* reveal votes, as noted by Ryan in [Rya10]. The fact that an unmarked ***freshly generated*** ballot paper reveals future votes in Prêt à Voter runs contrary to expectations provided by current voting schemes, which only serves to exacerbate this security issue.

The addition of printing on demand and re-encryption before scanning increases the distinct states into which Prêt-à-Voter ballot papers fall. There are two new states: ***proto-ballots*** and ***separated-mixed***. ***Proto-ballots*** contain only a seed which will be used to deterministically generate the ballot. The process by which this ballot paper generation occurs uses trapdoor information; without this trapdoor, the ***proto-ballots*** leak no information. ***Separated-mixed*** ballots are the result of applying ***separated*** ballots to one or more re-encryption mixers. The ***separated-mixed*** ballots have the same receipt freeness property as ***separated*** ballots; they also have the additional property that the ciphertext containing the candidate order does not match that generated by the printer, since the ciphertext has been re-encrypted. This prevents deanonymisation of the voters by the printer and its trapdoor without the assistance of all re-encryption mixers.

Candidate order size attacks

The size of the set of candidate orderings poses a (potential) problem for Prêt à Voter [RT09]. If the set of candidate orders is larger than the number of voters, then a coercer can request the voter cast a somewhat unlikely candidate order as their ballot paper which will then, likely, uniquely reveal their vote and act as a receipt. Due to its historical usage in Italy, this is referred to as an Italian attack. If the voter lies, there is little chance that the fake candidate order will actually have occurred. A potential solution to this is presented in [RT09]. However,

the solution presented only increases the number of candidates required for this to become a problem, without actually solving it. This issue has been resolved by re-permuting the candidate order while the ballots are being mixed, by the mixnet, after casting [BCH+12a].[4] Our Prêt à Voter variants incorporate this solution.

Print-on-Demand The use of in-polling booth ballot paper generation is necessitated in a significant number of government elections due to accessibility legislation, which allows voters to cast their vote at polling places other than those in their home district. A recent paper [CHJ+13] addresses printing on demand for Prêt à Voter.

Our approach draws upon [CHJ+13], targetting higher privacy levels. One of the primary conflicts raised in [CHJ+13] is between forward secrecy and auditability of printers; it is resolved in our approach by the additional re-encryption process, (by) removing the requirement of forward secrecy and leaving only auditability.

Attacking the Privacy of Prêt à Voter with control of the printer An entry on the bulletin board of a Prêt à Voter implementation consists of the pair (ranking, candidate-order ciphertext). This same information is mirrored on the receipt. This does not break privacy since the ranking (or selection) without knowledge of the candidate order could encode any vote. However, it is not necessary to hold the decryption key to recover the plaintext candidate-order. Any party could make guesses as to the candidate order and corresponding randomness then check against the BB. This attack is prevented in general by the size of the randomness. However the information printers learned when they created the ciphertext candidate-orders, namely the randomness, allows them to easily execute the aforementioned attack. Our approach is specifically designed to resolve this issue.

3.3 Overview of Our Solution

The core idea behind our solution is simple and practical. In this section we will explain the core technique behind our two proposed variants to Prêt à Voter, however, for the details of how these work in the wider context of the election process the rest of the chapter is required.

[4] Since the voter's candidate order will not appear on the BB, the attack detailed in Ryan et al cannot be executed. The attack fails because part 2 of the attack in the original paper is now impossible.

Our primary technique is to re-encrypt a ***separated*** ballot paper, either before or after marking, inside the polling booth to prevent single points of failure for privacy.

We can do this because Prêt à Voter, among others, has a ballot paper which contains a plaintext list of candidates, an encrypted component (from which the plaintext list could be re-constructed), and a space for voter input. The reason we need to re-encrypt is that an attacker who compromises the printer could otherwise break privacy; by using the unchanged encrypted component to correlate the plaintext list of candidates learned through printing and the voter's selection that will later appear on the Bulletin Board.

Throughout the rest of this chapter, we will describe the encrypted component as $E(p, r)$, the encryption of the permutation p using the randomness r. The permutation p here is the order in which the candidates appear on the ballot paper. Additionally, we denote the input component (voter's choice) as R, the ranking of the candidates according to permutation p.

In both cases the notion is a simplified representation of the way in which the ballot is constructed. The ranking R here should not be taken to imply that our method only works for preferential elections; our system works for most/all voting methods. For instance, in the simpler case of plurality voting the ranking will consist of a 1 in a given location and a null character everywhere else. Ranking here is synonymous with selection, choice, etc. This works for Single Transferable Vote (STV), Instant Runoff Voting (IRV), Plurality, Condorcet, Borda count, and Range, among others. Our point being that for concreteness, we focus on Single Transferable Vote (STV) and Instant Runoff Voting (IRV), although to the best of our knowledge there is no standard voting method to which our method would not apply.

In the new notation, our problem can be restated as follows: the printer learns the encrypted ciphertext-ids and the permutation $(E(p, r), p)$, and the scanner learns encrypted ciphertext-ids and ranking $(E(p, r), R)$; this allows them to correlate p and R, breaking privacy.

The goal of our mixing technique is simple: it attempts to take two sets, representing a ballot appearing to the adversary at two different stages (first through the printer, and second through the scanner or Bulletin Board), and make them un-connectable. The sets are $(E(p, r), p)$ and $(E(p, r'), R)$ where any change must preserve the relation between R and p. The commonality between the sets is $E(p, .)$ which provides the intuition that it is this data that must be mixed (re-encrypted). Indeed, both our variants primarily function by re-encrypting $E(p, r)$, albeit composed to achieve security in significantly different ways.

First Variant The first variant uses re-encryption to distribute the ballot generation and the entropy therein. Once the re-encryption has occurred the voter fills in their ballot paper. The difficulty of the filling process is captured by the mental calculation assumption. Privacy here is provided since no number of devices, less than the threshold, can learn the value of p; this directly implies information-theoretic privacy.

Second Variant The second variant targets the situation where an Electronic Ballot Marker (EBM) is provided for accessibility reasons. Re-encryption will seek to disconnect the identity of the voter from the information collected by the EBM, after the user has interacted with the EBM to fill their vote. Privacy here is provided because the EBM cannot connect any vote to any voter with better success than a generic passive Italian attack[5], without controlling all mixers. The nature of a polling booth that prevents the trivial revelation of the voters' identity to the EBM is captured by the assumption of a device-anonymous polling booth.

3.3.1 Differences Compared to a Mixnet

Our proposal has a strong similarity to a standard re-encryption mixnet; for that reason, we will in this section discuss some key differences.

In a standard mixnet the ballots are collected then processed as a series of batches, re-encrypting and permuting the ballots within each batch. In comparison, in both of our variants the voter directly takes their individual ballot paper to some or all of the re-encryption mixers, to permute the order of candidates within the ballot. If a standard method of mixing was used, the last mixer would have full knowledge of which ballot belonged to whom.

The possibility of timing attacks opened by the individual rather than batch processing has no effect on our first variant. In the second variant, a timing attack may allow a complete or partial identification of the voter, based on three-way collusion between EBM, scanner, and polling-station attendants in charge of ushering in the voters (or an onlooker with a facial recognition database). This would violate our Assumption 3 of a device-anonymous polling booth. This attack works on all schemes that rely on a device-anonymous polling booth.

The attack is made somewhat harder, or at least not easier, in our scheme since the data

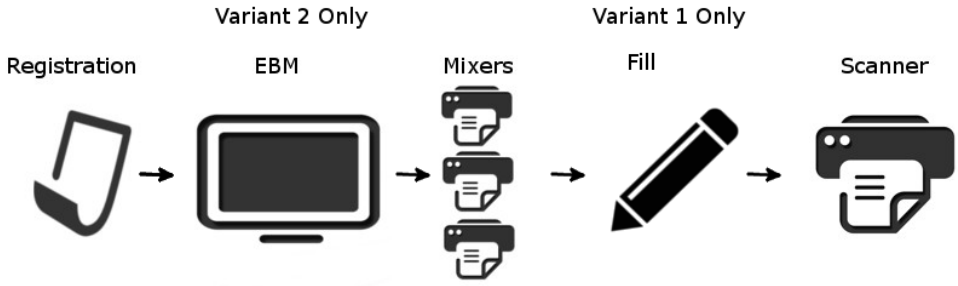
[5]The Italian attack works in systems where the set of all ballots cast is known; the attacker instructs the voter to cast an unusual vote and then checks to see if this vote occurs in the set of ballots cast.

appearing to the mixers does not allow corroboration of partial information from a timing attack. Despite this, in the case of our second variant, the full implications of requiring a device anonymous polling booth makes a strong contrast to a standard mixnet.

3.4 Variant 1: Human-Computable Permutation

In this section we detail the human-permutation variant, and the common parts with the EBM variant. The in-booth flows for both are depicted on figure 3.2.

Figure 3.2: Overview of the in-booth voting flow. The EBM step occurs only in Variant 2; the Fill step occurs only in Variant 1.



In our first variant we avoid providing the ranking R to any of the mixers. In a sense we use them in a similar way to the re-encryption clerks proposed by Ryan [Rya10]. In Ryan's solution, two separate ciphertexts, called *onions*, $E(p, r)$, are constructed: one is used with the cast ballot and another is extracted in the polling booth. In our variant the one onion is constantly updated with a new permutation. This update to the permutation is reported to the voter who then uses this knowledge to vote. In both cases the clerks or mixers are used to distribute the generation of the ballot paper which the voter will then fill in.

Ryan's solution does not provide privacy in Variant 1's model, because the device which makes the decrypted onion available to the voter learns the permutation under which the voter will cast their ballot. It can then match this information with the ranking that will later appear publicly, to break privacy. While Ryan's solution would work in Variant 2's EBM model, it may still allow an adversary knowing the initial permutation to trace votes, and it would require a device capable of *decrypting* the onions inside every polling booth: a risky proposition. If the candidate permutation size is sufficiently small as to not present a risk to privacy—and other security risks are deemed acceptable—then the useability advantages of Ryan's solution make it preferable to our solution here.

The various agents and components of our system are:

1. **An Election Authority (EA)**, tasked with running the election, and controlling:
   (a) **Printers**, to print freshly generated ballots,
   (b) (in Variant 2 only) **Electronic Ballot Markers (EBMs)**,
   (c) **Scanners**, to record the final separated-mixed ballots;
2. **Election tellers**, typically interested parties such as political parties, and here used as members of the privacy threshold, controlling:
   (a) **Mixers**, for mixing single ballots inside the polling booth,
   (b) **Mixnets,** for anonymising batches of ballots on the BB before tallying;
3. **A Bulletin Board (BB)**, realising a broadcast channel with memory;
4. **Ballot Generators**, generating (the randomness used for) fresh ballots.

It is expected that ballot generators and election tellers will be the same entities, and that the printer will print the ballot papers as generated (this is easy to verify, see below).

Ballot Construction

For convenience we assume that ballots are generated in the form suggested in [BCH+12b]—but without ballot packing[6] (this makes mixing of the encrypted permutation and plaintext ranking easy). That is to say, the onion (or encrypted ballot) is a tuple of encrypted candidate-IDs; for example, in an election for parties $\mathcal{A}, \mathcal{B}, \mathcal{C}$ the onion will be the concatenation of $E(\mathcal{A}), E(\mathcal{B}), E(\mathcal{C})$.

Audit 1: Checking ballot construction The most common suggestion to realise in-booth ballot construction is to use a trapdoor Verifiable Pseudo-Random Number Generator (VPRNG). A Verifiable Pseudo-Random Number Generator (VPRNG) allows verification that the Pseudo-Random Number was correctly generated from a given seed, but prevents evaluating the generator without knowledge of the trapdoor. These are similar to VRFs [MRV99]. The output of

[6] Ballot packing allows multiple candidates to be stored (packed) inside one ciphertext.

VPRNG can be used as input to a function which generates ballots. Once ballots are generated they can be printed on paper. Verification of correctness of the VPRNG, the correct running of the ballot generation function, and the correct printing of this information forms a valid proof of ballot construction. A simple case to consider is when a non-randomised signature scheme is used as a VPRNG. The auditor checks the validity of the signature, then runs the public function on the signature and checks that the ballot printed matches the output of the function. The audit can be conducted on any machine, since there are no privacy implications. To perform an audit the voter checks the following:

1. The public key printed is as expected, namely valid for the printer.
2. The signature is valid on the serial for that public key.
3. The rest of the content of the ballot is the correct output of the publicly known function on that signature.

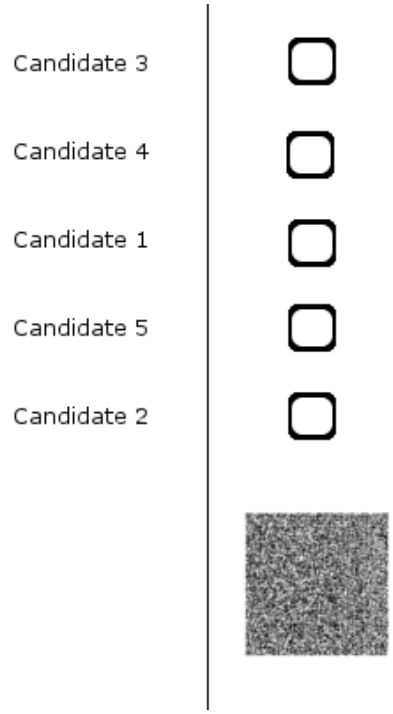
3.4.1 Voter Journey

The set of devices involved in Variant 1 is shown in Figure 3.3.

Figure 3.3: Overview of Variant 1.

Check-in with Election Officials The voter enters the polling station and registers with officials. At this point the voter is given a ballot paper by the official, see figure 3.4. This is similar in process to that provided by the printer in [BCH+12b], with the notable difference that we require the entire ciphertext to be printed on the ballot to enable mixing and re-encryption, rather than a mere serial number. The voter can optionally choose to audit the ballot paper, as detailed in Audit 1, without invalidating it.

Figure 3.4: Ballot paper ready to be voted on.



Third-Party Mixing (Optional) The voter physically separates the Left Hand Side (LHS) from the Right Hand Side (RHS). The voter is then allowed to have their RHS re-permuted and re-encrypted by any sequence of the mixers provided. If the voter chooses to have their ballot mixed, they input their ballot into the mixer. The mixer reads the ballot, permutes and re-encrypts the onion. Upon completion the mixer outputs its change to the permutation and a new RHS, which the voter can either further mix or cast their vote on. In addition, the mixer outputs a receipt which the voter takes home to verify that the mix was done correctly, as per Audit 2.

Audit 2: Correct mixing As previously mentioned, the mixer prints an audit paper with the new permutation and randomness values revealed. The voter can take this paper home to verify the correctness of the mix. This audit paper should be signed by the mixer to provide non-repudiation.

To verify the i-th round mix, the voter checks the following:

- The ciphertext candidate-IDs output by the mix are equal to the candidate-IDs re-encrypted with the randomness and permutation claimed;
- The ranking output is equal to the ranking input permuted with the claimed permutation.

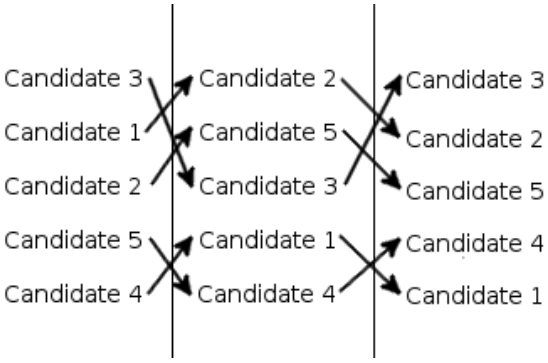


Figure 3.5: Visually composing a sequence of permutations, from (3,1,2,5,4) to (2,5,3,1,4) then (3,2,5,4,1). The composed permutation is the elements of the second permutation permuted by the order of the first.

Filling The voter composes the LHS of the original ballot with the permutations of each subsequent mix, in sequence. This can be done by placing the printouts next to each other and updating the permutation as shown in figure 3.5. Once the final permutation is calculated, the RHS can be filled in. The voter must then discard the LHS of the original ballot paper.

Scanning With the final ballot—suitably re-mixed to the desired "privacy threshold"—now filled in, the voter then goes to the scanner. The scanner submits the ballot to the Bulletin Board (BB), which sends back a signature on the hash of the ballot. The scanner prints out this signed hash which the voter can check later on BB. If the ballots used were not kept private before the election, the scanner should perform an additional mix before scanning and uploading.

Audit 4: Mix correctness [7] To verify the i-th round of mixing, the voter checks the following: The ciphertext candidate-IDs outputted by the mix are equal to the candidate-IDs re-encrypted with the randomness and permutation claimed.

Audit 5: Signature correctness The signature can be checked by an external computation device. The voter would be permited to use these devices outside the polling booth only.

Audit 6: Receipt on Bulletin Board If the receipt does not appear on the BB then the voter can produce their RHS. The presence of a valid signature on the receipt is considered proof that it should have appeared on the BB.

Mixing and Tallying Since, from the viewpoint of the authority, our ballots are constructed in the same way as in the Victorian elections using Prêt à Voter [CRST15b] (albeit without

[7] Audit 3 is temporarily omitted; it will be needed in the second variant of the scheme.

vote packing), the methods presented in that paper, and indeed any appropriate method from literature can be applied. The details are omitted for brevity.

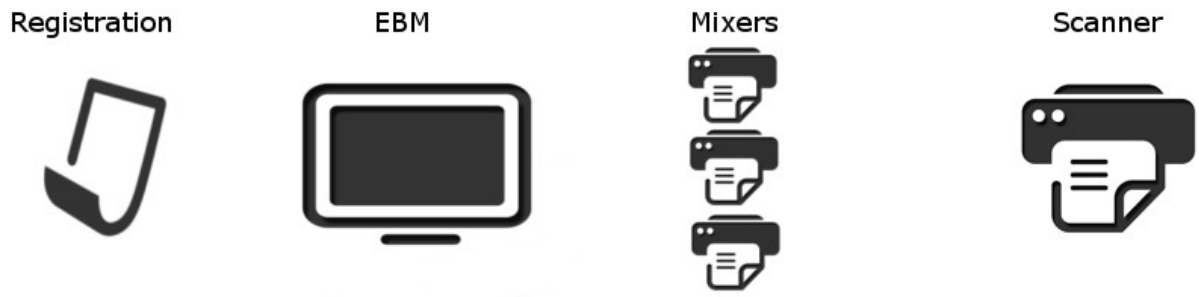
3.5 Variant 2: EBM-Assisted Variant

In our second variant we include an Electronic Ballot Marker (EBM) to assist the voter in filling their ballot—which in doing so necessarily learns the votes. Under the Device-Anonymous Polling Booth assumption, however, the EBM will not directly learn the identity of the voters casting those votes. Unfortunately, if the ballots were then immediately scanned and posted without further mixing, the $E(p, r)$ value seen by the EBM would become public and linkable to a specific voter, under the Public-Receipts assumption. The process of re-encryption mixing in Variant 2 follows almost directly from standard mixnets.

3.5.1 Voter Journey

In contrast to Variant 1, the implementation of these steps must not contradict the device-anonymous-polling booth assumption. The set of devices involved is shown in figure 3.6.

Figure 3.6: Overview of Variant 2.



Election Official The voter enters the polling station and registers with officials. At this point the voter is given a ballot by the official which is similar to that provided by the printer in [BCH+12b]. The voter can undertake Audit 1 to check that the ballot is correct. An example ballot is shown in figure 3.4.

Filling The voter takes the ballot to the Electronic Ballot Marker (EBM). This ballot contains a plaintext version of the candidate list, a machine readable candidate list in the bottom left and

an encrypted candidate list in the bottom right.

The EBM transforms the candidate list into the standard order and displays the result on its touch screen. The voter can then enter their preferences in the standard manner. Once the voter has entered and confirmed their choices the EBM overprints them on the ballot paper.

Audit 3: EBM printing The voter should check that the selection printed on the ballot reflects their choices. At this point the voter is required to place their LHS into a disposal bin[8], to ensure receipt freeness.

Party Mixing (Optional) The voter is then allowed to have their ballot mixed by any subset of the mixers provided in-booth, in any order. If the voter chooses to have their ballot mixed, they input their ballot into the mixer. The mixer permutes and re-encrypts the onion (encrypted preferences), and effects the corresponding permutation on the ranking. Upon completion, the mixer outputs a new RHS, which the voter can either further mix or take to the scanner. In addition, the mixer outputs a receipt which the voter takes home to verify that the mix was done correctly.

It is important to note that the Right Hand Side (RHS) which is being mixed does not reveal the vote. If it did this process would allow the voter to prove their vote to any of the parties which are providing mixing. The full set of parties providing mixing cannot break privacy without the assistance of the election authority, nor can the election authority break privacy without the full set of parties.

Scanning As before, once the ballot has been sufficiently remixed to the desired privacy threshold against the election authority, the voter goes to the scanner which performs its own final mix. The scanner then submits the ballot to the Bulletin Board, which responds with a signature on the hash of the ballot. The scanner prints out this signed hash which the voter can check later.

[8] In standard Prêt à Voter the disposal bin needs to be a shredder which completely destroys LHS to prevent any future linking, contra our variant where this information is assumed to be public.

3.6 Practical Matters

One of the primary concerns with our improvement is an increase in time and complexity of the voting process, which has a cost. We argue that in general the cost of this improvement is less than the amount spent on printers, EBMs and scanners already. Since there are normally only a few major parties across a country and several smaller relevant parties in each electorate, the time spent at an EBM is orders of magnitude higher than that required to scan and mix a few times, even if all major parties were to offer a mixer. Since the number of mixers is small and the number of EBMs required is high, the relative cost seems reasonable.

3.6.1 Electronic Ballot Marker (EBM)

We suggest a touch-screen-based Electronic Ballot Marker similar in style to that suggested in "Using Prêt à Voter in Victorian State elections" [BCH+12c]. We would, however, like to see the use of an EBM be optional for voters concerned for their privacy and willing to undertake the necessary mental effort.

If in-booth ballot generation is desired then the EBM is also used for this additional step. This is easy, since in-booth ballot generation requires scanning, printing and computing, all of which the EBM can already do. The ballot-generation is audited by the process described above.

Guidelines

We detail several guidelines and improvements which a good EBM should follow and incorporate. Many of the improvements are not possible or useful without the in-booth mixing we presented earlier. They are, of course, largely procedural in nature and mitigate rather than eliminate risk.

Optional While a voter should have the opportunity to use an EBM to assist in ballot marking, the process should be optional. Voters unwilling to trust the EBM should have their wishes respected.

Offline The EBM should have the smallest possible output capabilities required to assist with ballot marking. The EBM should not be networked or be able to be networked; the only output functionality required is printing.

Stateless We suggest the EBM be built in a stateless manner, that is, with no secondary storage devices which can be written to. In practice a low-end PC booting from a read-only CD seems a relatively cheap solution. At the end of a polling session, the EBMs would be powered down and hence lose all data. This would prevent the EBM from recording ballots and reduce the difficulty of verification.

Minimal Ideally the EBM would be entirely mechanical. However, the difficulty of receiving and processing permutations and selections—as well as assisting disabled voters—creates novel challenges for mechanical engineering.

3.6.2 Auditing

The system provides five or six personal audits, depending on the variant, in order to ensure integrity. We summarise the overall audit flow as follows.

Audit 1: Correct Ballot Construction This first audit ensures that the original ballot correctly captures the voter input, i.e., that the claimed-printed permutation of the ballot is the same as the actual encrypted permutation.

Audit 3: Correct EBM This audit step, only used in Variant 2, ensures that the EBM correctly records the voter selection on the ballot.

Audits 2 and 4: Correct Mixing (Individual) Audits 2 and 4 verify the mixes before and after marking. The two are virtually identical, the one difference being that Audit 4 further requires that the ranking has been correctly updated.

Audit 5: Signature Correctness This audit involves checking that the scanner has read the ballot correctly, and properly committed to its receipt.

Audit 6: Ballot Collection This audit ensures that the ballot has actually been collected and will be counted.

The cumulative effect of this set of audits is to ascertain that the ballot correctly captures the voter's input (1,3); the ballot will not be changed (2,4,5); and the ballot will be collected and input to counting (5,6).

There are also two universal audits. We require each in-booth mixer to provide a universal proof of shuffle, in addition to its individual proofs. This can be achieved by using a [Nef01] style proof.

Audit 7: Correct In-Booth Mixing Universal This audit ensures that all in-booth mixers acted correctly on all inputs.[9]

Audit 8: Correct General Mixing Universal This audit ensures that all mix servers acted correctly on all inputs.

3.6.3 Interpreting and Remedying Audit Failures

The question of what to do when an audit fails is known to be non-trivial; we make some brief suggestions here. We first note that failure of an audit may fall into one of three general categories: 'spoof-able', 'manageable' and 'delayed'. We will explain these categories as we come to them.

One of the major issues with audits in electronic voting is the desire of disgruntled voters to cast false aspersions on the integrity of the election. Our scheme, along with many others, counters this issue by digitally signing all the receipts. This means that a disgruntled voter cannot produce a valid fake receipt without breaking the underlying signature scheme. However, this does not entirely resolve the issue, since a malicious device can now produce a receipt with an incorrect signature which when the voter complains would not be believed. This brings us to Audit 5 in our scheme where the voter checks the signatures on their receipts. This audit can be spoofed by voters wishing to cast doubt on the election result, albeit with significant difficulty. This makes the response to failures of this audit particularly difficult. The easiest solution is likely to be physical, complicating the receipt (by using non-standard paper and/or watermarking, among many other methods), which would make it harder for voters to spoof.

Having dealt with the issue of false aspersions, it is no longer possible for election trustees

[9] This audit is required to prevent a Pfitzmann malleability attack [Pfi95] which would break privacy.

to avoid culpability for their negligence or deliberate attacks. This provides a significant disincentive for parties to attack the system. The remaining audits fall into two further categories. Failure of Audit 3 is 'manageable' since it is detected before the voter has continued with their vote. The voter should then spoil their current ballot and vote again.

Audits 1, 2, 4 and 6 are in the 'delayed' category; that is, the results of delayed audits have the potential to be known only after the election has concluded. Failures in this category are problematic for election organisers. On one hand, it is possible to correctly ascribe which party caused the issue, which clarifies blame and allows the negligence of trustees in previous elections to be used as a factor in their future selection. On the other hand, by the time these results arrive the election may already have ended. If a significant number of audits fail the election authority has a problem without a clear remedy. If the affected votes are predominantly in a given contest or election booth it is possible to re-run the election only in the affected area. However, we note that in the case of widespread failure a total re-run of the election may be the only solution.

3.7 Conclusion

We propose a refinement of Prêt-à-Voter in-polling booth end-to-end verifiable electronic voting schemes, which provides privacy against *ad hoc* compromises of individual poll-site devices without dependence on prior secrets or personal trusted devices. In addition, our improvement alleviates the privacy issues of ballot generation and storage since their contents are no longer required to be secret. This greatly simplifies the pre-election logistics, while also solving the problem of forward secrecy and auditability of printers. Our solution relies on the use of autonomous individual-ballot third-party mixing and re-encryption inside the polling booth.

While the need for additional in-booth hardware may make it unsuitable for some voting scenarios, we contend that the benefits outweigh the costs. This is particularly so in situations where the voters are unwilling to trust the election authority, or fear reprisal for voting according to their conscience—an increasing concern worldwide. The method we present seems inapplicable to STAR-Vote or any scheme which uses direct encryption. It is an issue of on-going investigation whether Scantegrity II can be adjusted to have re-mixable confirmation codes, which would make it eligible for the privacy enhancement of our technique.

Chapter 4

VOTOR: Conceptually Simple Remote Voting against Tiny Tyrants

---

In digital era, privacy must be a priority. Is it just me, or is secret blanket surveillance obscenely outrageous?

(Al Gore)

Abstract

The work in this chapter seeks to address the need for fair elections in an adversarial, ad hoc, and online setting. A major issue with existing online solutions is reliance on authorities, which in practice are often instantiated by only one organisation. We propose a conceptually simple but highly robust approach for casting ballots over commodity anonymisers under minimal assumptions. While other schemes have followed a similar approach, none have utilised the primitives to achieve so many desirable properties.

We then exploit this to construct a practical instantiation, called VOTOR—Voting Over TOR(though one need not use Tor specifically)—whose properties we show to compare favourably with prominent modern remote voting schemes such as Remotegrity and Helios. In particular, it shares the same coercion resistance property, which we call weak receipt freeness, as the latter two. Contra the others it also provides privacy against election tellers. It is also conceptually simpler in design than most modern schemes, and can be instantiated to allow voting credentials to persist from one election to the next without privacy attacks on past elections.

4.1 Introduction

The electronic voting literature started with the general problem of simultaneously achieving privacy and integrity. Integrity is easy if we know every voter and their vote. Privacy is easy if we know nothing about the vote/voter relation. A mixnet—network of mixers—allows encrypted votes to be anonymised before decryption. In essence, a mixnet lets us go from knowing the voters but not the votes to knowing the votes but not the voters. A secure mixnet also provides proof that no votes were added, removed, or modified in the mixing. Mixnets are often used in electronic voting because they neatly solve this basic contradiction between privacy and integrity. However, as we shall see, the mixnet introduces its fair share of complications. An alternative method is to use homomorphic encryption to tally the votes before decryption, which provides better privacy, but is less flexible.

Security relies on the existence of at least one honest mixer. Mixers are often idealised as parties under independent control, which are further assumed to have disjoint interests. This assumption however has often been false in the real world, due to the difficulty of getting suitable parties to expend the resources to participate.

"Electronic Elections: A Balancing Act" [Rez10] provides a chilling account of why this assumption is wrong. It highlights the story of "the Old Republic" in Brazil which "was plagued by collusion." This collusion allowed one group to maintain ostensibly legitimate elections for decades. "Election organizers and two main political groups, led by landed gentries, were involved. Regardless of the real outcome, the books were cooked at each election so that the two groups would alternate at filling the country's presidency, while the three pretended, with help from the fourth power—mainstream media—to find no wrong through the electoral supervision process."

More recently, Australia joined the US-led was on Iraq despite 60% to 70% of polled voters opposing, with the political party then in opposition providing only token objections. The "world's greatest democracy" itself not only has been forced to admit to the use of torture and mass surveillance, it is continuing such unconstitutional practices under bipartisan support despite public outcry. In addition to the manifest trust issues with mixers in high stake elections, the practical issue of getting sufficient disjoint parties to participate cripples the feasibility and practicality of smaller scale schemes relying on mixnets.

Even in the case where mixers are rightly trusted, the device on which the voter encrypts their vote still learns the vote. Their are several solutions to this problem but by far the most popular is code voting. In code voting, the authority and the voter share a code sheet which can be used to send a vote through the computer without it learning the vote. Unfortunately, at present, all known code voting solutions conflict with independence from the authority.

Coercion resistance in the remote setting is desirable but difficult. In the in-polling booth environment, the controlled booth is relied on to prevent unmodeled leakage. With an uncontrolled environment this is not possible. The schemes that "prove" coercion resistance do so without capturing the idea that the voter may want to prove their vote, in which case all bets are off. It is not sufficient to define a model and process, and show that nothing in that model using that process breaks coercion resistance. A willing voter has many options to record the voting process—outside of its model—in an arbitrarily convincing way. Furthermore, the zero-knowledge proofs that are often used for credential distribution do not work in the remote voting setting. We detail several attacks on schemes claiming coercion resistance in our comparison and discussion (Section 4.6), though we note that these attacks do not break their security proof but rather demonstrate weakness in the models.

Lost in the academic debate on coercion resistance and device independence, the question of how to practically prevent wholesale fraud by the election apparatus itself is mostly left unanswered. Instead we have seen specific countermeasures to specific threats. For instance, the allegations of fraud that followed the CEO of a prominent American vendor of electronic voting equipment promising to "deliver"[1] the election to the president then in power, eventually led to a Californian audit of electronic voting machines to thwart device-based attacks. However, the issue of practically decentralising trust in the election apparatus itself to prevent these threats remains unresolved.

4.1.1 VOTOR

In this paper, we show how to use several recent cryptographic primitives to construct a particularly practical yet private system, which we call VOTOR—"Voting Over TOR" (or any other anonymous channel).

[1] For additional information please see the New York Times article "Hack The Vote" Paul Krugman (December 2, 2003).

Specifically, our system correctly leverages the power of Linkable Ring Signature (LRS) over commodity internet anonymous channels (such as "the onion router" TOR as one of many options), to achieve multiple significant properties of interest in remote voting, namely:

Privacy and integrity from the election authority First and most importantly, our system withholds from the election authority the knowledge required to construct a valid ballot. This prevents the election authority from stuffing the polls, e.g., by casting ballots of its choosing ostensibly for voters registered but not engaging in the election. In addition, this lack of information prevents the election authority from breaking voter privacy. While both of these are basic requirements, none of the modern schemes which we use for comparison achieves them both.

Robust anonymity Secondly, our system makes minimal assumptions on the anonymous channel, be it Tor upon casting and/or a mixnet later on, mitigating the typical Achilles' heel of computationally efficient remote voting. In particular, correctness of the channel is not assumed, and neither is verifiability (unlike typical requirements on mixnets). Indeed, any attempt by the anonymous channel to substitute or modify votes is bound to fail, as it would reduce to an attempt to forge signatures. As for the assumptions needed to ensure voter anonymity, we shall see that they are easily realised in practice, despite Tor's shortcomings.

Dual (redundant) privacy The system relies on inherently anonymous linkable ring signatures and low-cost anonymous channels for their transmission. The latter can be combined in arbitrary ways. For example by using two anonymous channels, chosen by the voter and the election authority, we create redundant guarantees where privacy is ensured if either channel provided privacy. More generally, the system provides privacy if either the voter or the authority acts correctly.

Voter control The voter can freely choose one or several (unaudited, unreliable, unverified) general-purpose anonymous channel(s) through which to submit their vote to the election web bulletin board. Provided that the election authority cannot break the anonymity of this channel, the voter achieves privacy against them.

Reuseable and revocable credentials The setup of a VOTOR election requires the authenticated registration of each eligible voter's public key for official publication in the voters'

roll prior to the election. It is indeed a requirement of our scheme, partly to ensure privacy, and partly to prevent ballot stuffing, that all votes be ring-signed with respect to the precise officially designated ring of eligible voters. Key registration only needs to be performed once per voter, and the keys will remain valid indefinitely as long as each election is assigned a unique identifier (to block double-voting prevention from spilling over from one election to the next). Compromised keys can however be revoked at any time by the voter, and new keys re-certified between elections.[2]

Forward secrecy The scheme provides forward secrecy. During an election the voter submits their ballot and then (privately) updates their credential. The updated credential can be used in future elections without any further communication with the election authority, whereas forward secrecy prevents the updated credential being used to unmask or trace submissions in past elections, even under coercion.

We note that, along with Helios and Remotegrity, the system we propose is only suitable for low-coercion elections, since it only achieves a weak definition of receipt freeness and no functional coercion resistance. Against determined coercers, in-booth voting has no substitute.

Yet, despite the limitations of remote voting, VOTOR's properties confer the ability to perform fair elections with remarkable efficiency, to the point of enabling secure polls and referenda on a weekly or daily basis—taking the notion of direct democracy to its internet-era logical conclusion.

4.1.2 Background

Our approach makes use of two relatively new innovations: linkable ring signatures and commodity internet anonymous channels.

Ring signatures [RST01a] allow signers to send authenticated messages without revealing their identity within an arbitrary "ring" of possible signers. This enables voters to submit their vote in the "ring" of voters—where no attacker can distinguish which voter submitted that vote. Linkable ring signatures [LW05] allow multiple messages generated by the same signer to be linked. There is a second group of signature schemes called group signatures which require a

[2] Voters not wishing to bear the useability costs of protecting their private keys can re-register before every election. This becomes a useability decision for the election authority and/or the individual voters, rather than a security issue.

group manager with anonymity revocation powers [CvH91] [CSST06], however the presence of such a group manager is fundamentally at odds with the privacy from election authorities which VOTOR achieves.

As a basis for comparison, we have chosen Remotegrity [ZCC+13] and Helios [Adi08] because they respectively represent strong candidates in device independence and practicality.

Commodity internet anonymous channels such as Tor are growing in the public consciousness. We make use of them to submit votes. Voting places harder requirements on Tor and other channels than they were designed for, however these limitations are mitigated by the submission protocol we detail. The submission protocol allows voters to achieve arbitrarily high levels of hiding, even if we assume that the election authority knows exactly where the voters' submissions are coming from.

We utilise these primitives to construct a simple yet powerful remote voting scheme, provided that one can ensure the validity of a handful of reasonable assumptions:

1. A full list of the eligible voters' public keys can be properly authenticated and published.
2. Signatures are unforgeable and anonymous within a "ring", except in the case of double signing.
3. Different signatures by the same signer can be reliably detected, either by linking or tracing.
4. There exists an (untrusted, unreliable) anonymous channel for submitting anonymous votes.

4.1.3 Motivation

While the cryptographic literature mentions applications to e-cash and e-voting as motivation for the development of linkable ring signatures, no evidence has been found—after an extensive search—that either application has been studied in any detail or in any rigorous way. Indeed, the remote voting literature remains seemingly untouched by this specialised line of cryptographic primitives.

In "Traceable Ring Signatures" [FS07] and follow-up work [Fuj12] the authors devote a paragraph to the application of voting. Due to extreme brevity, the exact details of what they had

in mind are hard to reconstruct. It seems to be designed to allow anonymous elections between ad-hoc users on internet forums. They seem to assume the existence of both an anonymous channel and a web forum. The forum is both authenticating enough to convince the participants that public keys being posted from individual users are legitimate, and yet anonymous enough to allow voting without de-anonymising. In a footnote they propose combining traceable ring signatures with a technique for achieving receipt freeness [Oka98b]. Such a system would be both receipt free and achieve privacy against the election authority. This would be a major breakthrough in voting, but it appears that the notion they consider is much weaker than the one offered in the present chapter. They also do not take advantage of the dual linking property of traceable ring signatures, hence their application could be as easily solved by a linkable signature [LWW04].

4.1.4 Contribution

As already noted, there is an increasing push towards remote voting schemes based on unrealistic coercion resistance models and/or the promising but currently—and possibly fundamentally—limited notion of device-independent remote voting. Both of these avenues ignore the very real threat posed by collusion and corruption by, and of, the authorities.

We make several original technical contributions:

1. An extended security framework expressive enough to fit VOTOR, Helios and Remotegrity.
2. A detailed submission protocol suitable for voting using weak internet anonymisers such as Tor.
3. A detailed construction of a remote voting scheme, VOTOR, based on LRS over Tor.
4. An analysis of the properties of this class of schemes under realistic assumptions.
5. A comparison to the current state of the art in remote voting schemes.

We show that, despite its conceptual simplicity, our scheme provides a significant and unique combination of benefits for remote voting, both in terms of security and practicality. Our scheme compares favourably to prominent remote voting schemes from the literature, such as Helios [Adi08] and Remotegrity [ZCC+13]. Our scheme is also superior to speculative constructions based on decentralised pseudonymous transaction platforms provided by cryptocurrencies.

A remote voting scheme based on the Bitcoin protocol [Nak08], for example, would use the near-anonymous pseudonymity and irreversibility of Bitcoin transactions to implement a kind of pre-election mixing, not of the ballots cast, but of the voting credentials. The election authority would initially issue voting credentials (encoded as special Bitcoins) to each voter. The voters would then mix them for anonymity, as described in [BBSU12], to achieve credential ownership privacy. Each of the resulting anonymous "bearer credentials" could then be cast into a vote by spending the coin over an anonymous network, such as Tor. The privacy of such a scheme relies on the amount of peer-to-peer mixing that must take place before the election, and the anonymity of the channel used to cast the vote during the election.

Compared to VOTOR, this back-of-the-envelope Bitcoin-based voting system requires the same assumption of anonymous channels. However, the lengthy pre-mixing required to achieve credential privacy is a clear drawback compared to VOTOR, which achieves privacy directly by cryptographic means. Lastly, Bitcoin votes are as irreversible as the Bitcoin transactions that buttress them, which encourages vote selling by preventing the revocation of compromised credentials. VOTOR, by contrast, discourages vote selling by implementing an effective revocation mechanism, as discussed in a later section.

4.1.5 Receipt freeness and Coercion-Resistance

The very definition of "receipt freeness" itself is not well accepted. We take the stronger definition from the original paper [BT94] based on the premise that, "If a voter is allowed, but not required, to keep a vote secret, the voter could be coerced by an employer or anyone with power over the voter into casting a certain vote." Indeed, since all voter actions can be observed with the voter's participation in a remote voting scheme, if the voter can be convinced that their vote is accurately recorded, then the coercer can be as well.

While several remote schemes claim to be coercion resistant, these claims normally require three highly questionable assumptions:

- Firstly, that there exists a trusted remote device which has the power to prove votes but never will.
- Secondly, that the coercer be absent from the voter at some point.
- Thirdly, that the voter have the mental capacity and readiness to act consistently towards

coercers.

The original definition of coercion resistance does not allow the voter to access computational devices or information channels to which the coercer does not have access. This appears stronger than the trending literature [DKR06]. Many of the remote coercion resistant voting schemes are perhaps better thought of as "voting extension" schemes. That is, if one already has a secure environment in which one could vote, then these schemes allow the use of that environment to securely vote later (outside of that environment). This is a highly valuable contribution but not a substitute for the in-polling booth secure environment.

Blind Signature Schemes

VOTOR is conceptually closest to the class of blind-signature-based voting schemes. The registration phase in VOTOR can be thought of as the issuing of blind signatures, and the voting stage as the submission of the signed vote on an anonymous channel.

We note that Okamoto's "Receipt-Free Electronic Voting Schemes for Large Scale Elections" [Oka98b] uses blind signatures and claims to be receipt free. Alas, Okamoto's scheme rests upon several assumptions, whose properties cannot be instantiated in the remote setting. We refer the reader to our discussion of JCJ/Civitas (Section 4.6.1) which relies on many of the same assumptions.

All of the blind signature schemes we encountered rely on the existence of a trustworthy election administrator or recorder. The security of VOTOR in the face of untrusted election authorities and tellers provides a fundamental privacy and integrity advantage over pure blind-signature based schemes.

Summary

To summarise, the voting literature has moved away from the early conceptual schemes based on cryptography, and toward operational procedures centered on the voting ceremony to achieve sought properties. However, recent advances in cryptography and network protocols suggest that the methodology of the pioneers might be worthy of a second look. Accordingly, our scheme shares with those early works the requirement of an anonymous channel, of which Tor (amongst other available alternatives) is now a practical and useable approximation. On the

cryptographic side, the availability of primitives such as linkable ring signatures, compared to blind signatures, should favorably impact the design of modern remote voting schemes based on cryptography.

4.2 Preliminaries

In the preliminaries section we will discuss our security definitions and the various components of our scheme.

4.2.1 Security Definitions

We present extended formalisations of the standard election scheme security properties. Our formalisations are based on [SB13][BCP+11][BCPW12]. Our extension provides greater flexibility of voting schemes, and models ephemeral data required to meaningfully discuss remote receipt freeness. It should be noted that, there are other definitions which are arguably more natural and stronger. However, the type of security definitions used here are widely cited and commonly accepted, and are used for that reason.

There is great difficulty in designing a set of security definitions which are broad enough to cover all voting systems. The issue is further complicated by the requirement that the definitions should be robust enough to imply security, but flexible enough to facilitate proofs. We aim for a less ambitious goal of defining the sets of information that exist in a Web Bulletin Board (WBB) centric scheme, and defining security properties and adversarial powers based on these sets. This means that our general definitions often discuss advantage without concretely instantiating it and uses general descriptions of functions to abstract over the actual process used. The result is that someone verifying the proofs will need to check not only that the proofs are valid but that the instantiation of sets and functions is valid. A standard way to invalidly model voting is to exclude information from the model which the adversary has access to.

We define our security by proposing games representing properties and adversarial powers.

Definition 1. ***(Election Scheme)*** *An election scheme $\mathcal{ES}$ is a tuple of efficient algorithms* ***(Setup, AddVoter, Run, Vote, BB, Revoke, Tally)*** *such that:*

- **param** $\leftarrow$ **Setup**($\boldsymbol{\lambda}$) is a probabilistic polynomial time (PPT) algorithm which, on input of

a security parameter $\boldsymbol{\lambda} \in \mathbb{N}$, outputs a vote space **m**, an empty voter list **VL**, an authority public key $\boldsymbol{pk_A}$ and private key $\boldsymbol{sk_A}$ pair. **m** and **VL** are sets.

- ($\boldsymbol{sk_i}$, $\boldsymbol{pk_i}$) $\leftarrow$ **AddVoter(param)** is a PPT algorithm which, on the input of a security parameter $\boldsymbol{\lambda} \in \mathbb{N}$, outputs a private/public key pair $(\boldsymbol{sk_i}, \boldsymbol{pk_i})$. It then adds $\boldsymbol{pk_i}$ to **VL**
- ($\mathcal{EI}$, **vl**, **bb**, **elec-param**) $\leftarrow$ **Run**(**param**, **vl**) is a PPT algorithm which, on input **vl**$\subseteq$**VL** and **param**, outputs an election id $\mathcal{EI}$, a list (here called ring) of voters **vl**, a bulletin board **bb**, and additional parameters **elec-param**. **bb** is a multiset.
- $\boldsymbol{b} \leftarrow$ **Vote**$(\boldsymbol{sk_i}, \boldsymbol{v}, \mathcal{EI})$ is a PPT algorithm which, on input of a voter secret key $\boldsymbol{sk_i}$, a vote $\boldsymbol{v} \in \boldsymbol{m}$ and an election id $\mathcal{EI}$, produces a ballot $\boldsymbol{b}$.
- **accept reject** $\leftarrow$**BB**(**bb**,$\boldsymbol{b}$) is a PPT algorithm which, on input of a bulletin board **bb** and a ballot $\boldsymbol{b}$, returns **accept** or **reject**. If **accept**, **bb**$\leftarrow$**bb**$\cup\{\boldsymbol{b}\}$.
- **accept reject** $\leftarrow$**Revoke**(**bb**,$\boldsymbol{r}$) is a PPT algorithm which, on input of a bulletin board **bb** and a revocation request $\boldsymbol{r}$, returns **accept** or **reject**. If **accept**, **bb**$\leftarrow$**bb**$\cap\{\boldsymbol{b}\}^C$, where $\boldsymbol{b}$ is the ballot defined by $\boldsymbol{r}$.
- **V** $\leftarrow$**Tally**$(sk_A,$ **bb**) is a PPT algorithm which, on input of a bulletin board **bb** and authority private key $\boldsymbol{sk_A}$, outputs a multiset **V** representing the election result.

Ephemeral Data In addition to the information provided above we include the more abstract notation of an ephemeral dataset E, where $e_i \in E$ is the subset of ephemeral data regarding voter v_i. This notion seems essential to simultaneously discuss verifiability, receipt freeness and coercion resistance. If receipt freeness and coercion resistance are under discussion, ephemeral data should be very carefully instantiated to model all relevant data that the voter could possibly keep either through action or inaction.

Time Our definitions of a voting system also make no mention of time points. Any forward secure scheme would normally use time points to allow modelling of the forward security properties. In our analysis we treat each election as a time point and exploit forward security to reduce the burden on key registration between elections.

Oracles We consider the following oracles which model the powers of the adversary in attacking the system.

- $pk_i \leftarrow \mathcal{JO}(\perp)$. The Joining Oracle, on request, adds a new user to the system by calling **AddVoter**. It returns the public key pk_i of the new user.
- $b \leftarrow \mathcal{VO}(pk_i, v, \mathcal{EI})$. The Voting Oracle, on input: a public key pk_i, vote v and election identify EI calls **Vote** and returns **b**.
- $sk_i \leftarrow \mathcal{CO}(pk_i)$. The Corruption Oracle, on input of a public key pk_i that is a query output of $\mathcal{JO}$, returns the corresponding secret key sk_i.

We model key properties as games in this framework:

Privacy Intuitively an election system ensures privacy if an adversary learns nothing more about a voter's vote than that it is found in the final set of all votes.

More formally, an election scheme $\mathcal{ES}$ satisfies privacy if either of the following holds:

Voter indistinguishability Given **bb**, an adversary's advantage in guessing the voter v_i is negligible:

This is modeled by a game between a Simulator $\mathcal{S}$ and Adversary $\mathcal{A}$ in which $\mathcal{A}$ is given access to the oracles $(\mathcal{JO}, \mathcal{VO}, \mathcal{CO})$.

1. $\mathcal{S}$ generates and gives $\mathcal{A}$ the system parameters **param**.
2. $\mathcal{A}$ may query $(\mathcal{JO}, \mathcal{CO})$ according to any adaptive strategy.
3. $\mathcal{A}$ sends **vl**$\in$**VL** to $\mathcal{S}$. $\mathcal{S}$ calls **Run**(**vl**) and sends $\mathcal{A}$ ($\mathcal{EI}$, **vl**, **bb**, **election-param**).
4. $\mathcal{A}$ may query $(\mathcal{VO}, \mathcal{CO})$ according to any adaptive strategy.
5. $\mathcal{A}$ gives $\mathcal{S}$ $v' \in m$, the target vote, along with ballots $\{b_1, ..., b_i\}$ where i is the number of calls to the $\mathcal{CO}$ and the $\mathcal{VO}$ oracles.
6. $\mathcal{S}$ chooses a voter $\pi \in_R$**vl**, where π has not been an input to $\mathcal{CO}$ or $\mathcal{VO}$. $\mathcal{S}$ now runs **BB** on all bs provided by $\mathcal{A}$. It runs **BB**(Vote(sk_π, v')). Finally **bb** is given to $\mathcal{A}$.
7. $\mathcal{A}$ outputs a guess $\pi' \in \{$**vl**$\}$.

We denote by

$$\mathbf{Adv}_{\mathcal{A}}^{VoterInd}(\lambda) = |\Pr[\pi' = \pi] - \tfrac{1}{n-i}|,$$

where i is the total number of calls to $\mathcal{CO}$ and $\mathcal{VO}$.

Definition 2. ***(Voter Indistinguishability)*** *An election scheme $\mathcal{ES}$ is voter indistinguishable if for all PPT adversaries $\mathcal{A}$, $\textbf{\textit{Adv}}_{\mathcal{A}}^{VoterInd}(\lambda)$ is negligible.*

Vote indistinguishability An adversary's advantage in guessing the encoded vote v for a given $bb_i \in BB$ is negligible.

This is modeled by a game between a Simulator $\mathcal{S}$ and Adversary $\mathcal{A}$ in which $\mathcal{A}$ is given access to the oracles $(\mathcal{JO}, \mathcal{CO})$.

1. $\mathcal{S}$ generates and gives $\mathcal{A}$ the system parameters **param**.
2. $\mathcal{A}$ may query $(\mathcal{JO}, \mathcal{CO})$ according to any adaptive strategy.
3. $\mathcal{A}$ sends **vl**$\in$**VL** to $\mathcal{S}$. $\mathcal{S}$ calls **Run**(**vl**) and sends $\mathcal{A}$ ($\mathcal{EI}$, **vl**, **bb**, **election-param**).
4. $\mathcal{A}$ gives $\mathcal{S}$ ballots $\{b_1, ..., b_i\}$, where i is the number of calls to the $\mathcal{CO}$, along with a set V of $\{v_i \in m\}$ for all non-corrupted voters.
5. $\mathcal{S}$ chooses a voter $\pi \in_R$ **VL**, where π has not been an input to $\mathcal{CO}$. It also chooses an element $v \in_R$ **m**. $\mathcal{S}$ now runs **BB** on all bs provided by $\mathcal{A}$. It runs **BB**(Vote(sk_π, v)) and **BB**(Vote$(sk_i, v \in_R V)$) for all uncorrupted voters other than π. **bb** and π is given to $\mathcal{A}$.
6. $\mathcal{A}$ outputs a guess $v' \in V$.

We denote by

$$\textbf{Adv}_{\mathcal{A}}^{VoteInd}(\lambda) = |\Pr[v' = v] - \tfrac{1}{|\mathbf{m}|}|.$$

Definition 3. ***(Vote Indistinguishability)*** *An election scheme $\mathcal{ES}$ is vote indistinguishable if for all PPT adversaries $\mathcal{A}$, $\textbf{\textit{Adv}}_{\mathcal{A}}^{VoteInd}(\lambda)$ is negligible.*

Individual Verifiability (Collected as Cast) An election scheme $\mathcal{ES}$ satisfies individual verifiability (collected as cast) if:

For a given voter (pk_i, sk_i) the adversary cannot produce $bb_i \in$ **bb** corresponding to that voter without knowing sk_i.

1. $\mathcal{S}$ generates and gives $\mathcal{A}$ the system parameters **param**.
2. $\mathcal{A}$ may query $(\mathcal{JO}, \mathcal{CO})$ according to any adaptive strategy.
3. $\mathcal{A}$ sends **vl**$\in$**VL** to $\mathcal{S}$. $\mathcal{S}$ calls **Run**(**vl**) and sends $\mathcal{A}$ ($\mathcal{EI}$, **vl**, **bb**, **election-param**).

4. $\mathcal{A}$ may query $(\mathcal{VO}, \mathcal{CO})$ according to any adaptive strategy.

5. $\mathcal{A}$ returns b.

$\mathcal{A}$ wins the game if:

i. **BB**(**bb**,b) returns **accept**.

ii. b is "independent" of all calls to $\mathcal{CO}$.

iii. b is not the output of $\mathcal{VO}$.

We denote by

$$\mathbf{Adv}_{\mathcal{A}}^{IndColl}(\lambda) = |\Pr[\mathcal{A} \text{ wins the game}]|.$$

Definition 4. ***(Individual Verifiability (Collected As Cast))*** *An election scheme $\mathcal{ES}$ is individual verifiable (collected as casted) if for all PPT adversary $\mathcal{A}$, $\mathbf{Adv}_{\mathcal{A}}^{IndColl}(\lambda)$ is negligible.*

The former definitions are concrete and we will use them in the analysis of our scheme. In contrast the following definitions are more abstract. They provide an intuition for how we believe the properties should be defined without instantiating all the details.

Individual Verifiability (Cast As Intended) An election scheme $\mathcal{ES}$ satisfies individual verifiability (cast as intended) if: The adversary cannot produce a valid tuple (bb_i, e_i, v_i) where bb_i does not encode v_i with non-negligible probability.

Universal Verifiability (Counted As Collected) An election scheme $\mathcal{ES}$ satisfies universal verifiability (counted as collected) if: Given BB the adversary cannot produce a valid V where V is not the collection of all (and only) v_is which $bb_i \in BB$ encoded.

Weak Receipt freeness An election scheme $\mathcal{ES}$ satisfies weak receipt freeness if: If once all e_i have been destroyed, the adversary's advantage in guessing the encoded vote v for a given $bb_i \in BB$ is negligible.

Receipt freeness An election scheme $\mathcal{ES}$ satisfies receipt freeness if: If an adversary's advantage in distinguishing (v_0, e_0, bb_i) from (v_1, e_1, bb_i) is negligible.

Coercion Resistance An election scheme $\mathcal{ES}$ satisfies coercion resistance if: An adversary directly observing all of e_i, with access to bb_i, cannot tell if it encodes v_i.[3]

[3]Individual Verifiability (Cast As Intended) (where bb_i and e_i uniquely determines a v_i) is in direct conflict with this definition of Coercion Resistance. This forms an impossibility result. This result is avoided in in-polling booth schemes by the assumption of a polling booth which prevents direct observation of all e_i.

Adversarial models

We describe several adversary models which provide additional granularity to the games given above. Note that these models should not be thought of as disjoint, as an adversary could have several of these powers.

Passive Adversary The passive adversary has access to BB, V.

Corrupt Device The corrupt device can observe (some subset of) e_i. It can also change the e_i which is presented to the voter.

Corrupt Election Authority The corrupt authority can observe (some subset of) e_i. It can also change the e_i which is presented to the voter.

Corrupt Election Tellers The election tellers have access to the decryption keys to allow tallying. For this reason it can calculate v_i for any bb_i.

4.2.2 Cryptographic Primitives

A forward secure linkable (or traceable) ring signature is defined as a tuple of algorithms $\sigma = (\boldsymbol{Setup}, \boldsymbol{Gen}, \boldsymbol{Sig}, \boldsymbol{Ver}, \boldsymbol{Link}, \boldsymbol{Update})$.

1. **Setup**: On input security parameter $\boldsymbol{\lambda}$, outputs public parameters **param**.
2. **Gen**: On input public parameters **param**, outputs a signing/verification key pair $(\boldsymbol{sk}, \boldsymbol{pk})$.
3. **Sig**: Given a signing key $\boldsymbol{sk_i}$, $i \in N$, a tag L = $(\boldsymbol{issue}, \boldsymbol{pk_N})$, and a message $\boldsymbol{m} \in \{0,1\}^*$, outputs a signature σ.
4. **Ver**: Given a tag L = $(\boldsymbol{issue}, \boldsymbol{pk_N})$, a message $\boldsymbol{m} \in \{0,1\}^*$, a set of public keys $\boldsymbol{pk_N}$, and a signature σ, outputs a bit which is either "valid" or "invalid".
5. **Link**: Given a tag L = $(\boldsymbol{issue}, \boldsymbol{pk_N})$ and two valid message/signature pairs $(\boldsymbol{m}, \sigma), (\boldsymbol{m'}, \sigma')$, outputs one of the following: "independent," or "linked". If the signature is traceable then **Link** may also return "pk_i" under some circumstances.
6. **Update**: Given a signing key $\boldsymbol{sk_i}$, returns a new signing key $\boldsymbol{sk_i'}$ and optionally a new public key $\boldsymbol{pk_i'}$.[4]

[4]We present in Chapter 5 the first forward secure linkable ring signature. A forward secure linkable signature scheme can also be trivially constructed from any linkable ring signature, such as [LASZ14], by generating a new key and sending it with a signature using the old key. The old key must then be deleted. However, the latter approach requires a significant increase in communication and is less elegant.

Here, pk_N refers to the set of all public keys included in the ring of signers, and $issue$ denotes an arbitrary string indicating what the signature is about (such as an election identifier).

The linkable ring signature scheme should satisfy the following properties:

1. Tag-Linkability: If the same pk_i outputs valid $(m, \sigma), (m', \sigma')$ on the same tag L = $(issue, pk_N)$, then **Link** will return "linked". If the signature is traceable the public key will be returned.
2. Anonymity: Given access to three oracles Sig_{sk_0}, Sig_{sk_1}, Sig_{sk_b} where $b \in \{0, 1\}$, an adversary's advantage in guessing b is negligible.
3. Exculpability: No coalition of corrupt signers can produce a signature on behalf of another signer.
4. Forward Secrecy: No adversary can forge a valid signature from a previous time-frame even with knowledge of all current keys.

Instantiations of Linkable and Traceable Ring Signatures

The initial realisation of TRS from [FS07] operates in Diffie-Hellman groups and has a straightforward implementation. Their scheme is, however, rather inefficient with signature sizes in $\mathcal{O}(N)$ for a ring of size N. This restricts their use to small electoral precincts, or requires precincts to be further divided up into smaller parts to the detriment of their constituents' privacy.

A better choice is to use Fujisaki's sublinear TRS [Fuj12], which admits signatures of size $\mathcal{O}(\sqrt{N})$ for rings of size N. This makes online voting and verification practical even in precincts with thousands or possibly millions of voters. The main drawback of the Fujisaki scheme is that it uses bilinear maps over groups of composite order n with unknown factorisation. This requires a one-time trusted setup phase where appropriate bilinear groups are generated from a secretly chosen modulus $n = pq$, whose prime factors are then positively destroyed. This one-time setup can be done in a distributed manner using multi-party computation, to prevent any one party from ever learning the factorisation. Although knowledge of the factorisation does not allow one to violate the anonymity of Fujisaki's ring signatures, it allows one to create untraceable (or wrongly traceable) signatures.

Efficiency is also an issue for the tracing operation, which could act as a bottleneck and open

the WBB to denial of service (DoS) attacks, since in principle tracing might require pairwise tracing across all pairs of valid signatures collected in a given precinct, with complexity $\Omega(N^2)$. Fortunately, in Fujisaki's scheme in particular, the tracing of N signatures already accepted as independently valid, is based on string equality and set membership tests as the only pairwise operations. That is, they can be grouped together using sorting, and then performed in a single batch over the entire ring at a total cost of $\Theta(N \log N)$ units of time it takes to parse one signature.[5]

We mention in passing the forward-secure traceable ring signature introduced in [HL07]. Unfortunately, though not mentioned in later literature, this scheme appears to be broken because a signer cannot sign without breaking the forward security of the signature scheme. In chapter five, we present the first forward secure linkable ring signature.

4.2.3 Anonymous Channel(s)

An anonymous channel is a communication channel where the receiver learns nothing about the sender, except that which is revealed by the message. Some anonymous channels are verifiable which prevent the insertion, substitution, and deletion of messages. These anonymous channels can, however, always perform a denial of service attack.

There are two notable differences between the anonymous channels required in previous work and what we suggest here. First, we do not require that the anonymous channels be verifiable. This avoids costly mixnets which (in some cases) allow some small subset of messages to be undetectably changed. Secondly, and capitalising on the first difference, we make optional use of two anonymous channels to provide a system which is robust if either of the anonymous channels is effective.

TOR Description

TOR, The Onion Router, is an online system of voluntary relays which aims to allow real-time anonymous communication over the web. Tor works by constructing and routing IP packets across a random path amongst the relays, encrypting the packet and its routing information in layers, not unlike the "onion" in a decryption mixnet.

[5]The efficient batch-oriented tracing in $\mathcal{O}(N \log N)$ we describe is not mentioned in the Fujisaki paper, but follows readily from the description of the tracing algorithm. The technical details are omitted.

Specifically, the user's Tor client chooses a circuit of successive relays, and encrypts the payload together with the destination address under the public key of the last relay (called the "exit node"). The resulting ciphertext—together with the address of the node it is destined to, in this case the exit node—is then encrypted under the previous node's public key, and so on successively in layers, with the outer layer corresponding to the first relay. The user, having constructed their onion, then sends it to the first relay.

Provided that at least one relay honestly discards the received routing information, the link between the user and the destination is not directly revealed. In theory, timing and size information could allow network analysis. However, they would also be easy to thwart for short communications, by stuffing the layers of the onion with padding to be discarded from one hop to the next. Nevertheless, in our analysis we propose additional countermeasures that voters can optionally adopt to increase privacy.

Whereas mixnet-based voting systems force voters to rely on a pre-determined set of mixers over which they have no control, Tor allows voters to dynamically select which and how many relays they will use to protect their privacy. Indeed, in VOTOR the voter is not even bound to Tor. Individual voters can use as many implementations of anonymous channels as they wish, in any order, and they do not need to be real-time. Provided that a validly signed vote arrives at the WBB in a timely fashion (before the end of the election), the vote will be recorded and counted.

Tor Validity and Analysis

Due to the way in which Tor would be used in our scheme its validity as an anonymous channel is non-trivial. Specifically, since Tor routes its packets in near real-time, the issue of timing attacks (not an issue with standard mixnets) should immediately come to mind. It may be more accurate to say that we believe an anonymous channel meeting our specifications can be constructed using Tor, than to say that Tor itself is an anonymous channel. The details follow.

Security Model Our threat model of Tor assumes that the adversary is able to monitor the data leaving the voter's computer. It is also able to directly observe data arriving at the election authority. We assume the adversary cannot directly control or observe the voter's computer.

Attack The primary problem we experience with Tor is actually one of its features: its low latency introduces powerful timing attacks. For instance if the arrival time is distributed uniformly between 0.5 and 1.5 seconds and there are 100 valid votes arriving each second, then the voter only obtains privacy within this set. The total size of the electorate may be around 3600 times the anonymity set, even if the election lasts for only one hour.

Active and Passive Mitigation The level of traffic present in the Tor network provides a level of mitigation against these attacks. The first mitigation a voter can make is to create extra traffic themselves on Tor across the election period; this greatly expands the anonymity set of the voter. This mitigation is notable in that it requires no modifications to existing Tor nodes.

If modifications were allowed, we suggest the creation of a new class of Tor nodes for significant elections by interested parties. This new class of nodes mirrors existing Tor nodes with the additional parameter of a random time delay. This random delay allows the voter's (possible) submission to correspond to a larger set of received votes. The voter could combine this new class of nodes with normal nodes by suitably constructing the Tor circuit.

Weaker Validity For a general class of elections for which Helios was originally designed, no changes to Tor are required to provide a fundamental privacy improvement. The catch-phrase "Trust no one for integrity, trust Helios for privacy" can be replaced with "Trust no one for integrity, trust that VOTOR cannot monitor your encrypted traffic". The setup cost is much lower than arranging a series of mixers/tellers.

4.2.4 Ledger of Voters

In practice it is expected that the ledger of voters and the web bulletin board would be managed by the same authority and hosted on the same site. However, we separate them for the purpose of clarity since they perform two different functions in this scheme. The ledger of voters is a list of voters and corresponding public keys in a given election. Note that one of the requirements of linkable ring signatures is that all public keys be known. The issue of how to authenticate users and their public keys is at the core of Public Key Infrastructure (PKI) and is not unique to this scheme.

Although we do not, in fact, need a full-fledged PKI in our system, we need a way to

authenticate voters and publish their keys. We suggest, in passing, two possible ways to authenticate voters. The first is to send the voter a credential allowing them to upload a public key. The second and more secure way is to authenticate the voter in person and then accept their public key. Since public keys can be re-used across multiple elections and individual keys can be revoked, this latter method could be done once as a form of voter registration.

4.2.5 Web Bulletin Board (WBB)

One of the advantages of our suggested protocol is that the validity of any potential entry to the WBB is universally verifiable. The effect of this is that voters do not need a signed receipt from the WBB as proof that their vote should be published.

The WBB should publish all signed votes it receives that meet the following criteria:

1. **The vote meets some specified markup.** The only requirement here is that the markup be published and unambiguous, so that the correctness of any vote is universally verifiable.
2. **The signature should be valid on the message.** That is Ver(m, σ) = 1.
3. **The signature should be independent of any previously posted.** Link((m, σ),(m_i, σ_i)), for all $i \in$ previous submissions, = "independent".

In addition, the WBB should publish signed revocation messages which allow voters to invalidate their vote, in the case of vote selling or device corruption. A simple way to do this would be to publish non-independent messages as a public way to revoke a key.

To ensure the integrity of the WBB, standard methods such as signatures on the content of the WBB should be employed. These allow trivial proofs in the case that cheating is detected, since only the WBB operators could have signed the content, and in doing so, they promised to upload it [CS14].

The computational complexity and storage requirements of the WBB are surprisingly light for some implementations of LRS and TRS. As already mentioned for the Fujisaki TRS, the length of the signatures can be sub-linear in the number of voters in a given precinct, which reduces the storage and transmission requirements significantly. In addition, the tracing operation does not require any cryptographic operations other than signature verification, only string comparisons. This allows tracing to be batched across the entire WBB, with a complexity of $N \log N$ in the number of voters.

4.3 Construction

An election using VOTOR occurs through the following steps. The election authority authenticates the voters' keys and prepares and publishes them on the ledger of voters. The voters construct a signature on their encrypted[6] vote using the published ring for their precinct and the correct election ID. The voters submit their signed encrypted votes over Tor or in any anonymous way of their choosing. The voters check that their vote appears on the WBB; if not, they can resubmit an identical copy (in fact, they are advised to submit many identical copies at different times, to thwart timing attacks). Once the voting period is over, the tellers run a simple non-verifiable mixnet. The final result can then be tallied or verified by anyone.

For the next election, the voters simply reuse their existing (updated) keys without having to create or register new ones.

4.3.1 Specification

Setup On input a security parameter $\boldsymbol{\lambda}$ the setup outputs an empty voter list **VL**, vote space m, authority keys $\boldsymbol{pk_A}$ and $\boldsymbol{sk_A}$ and **LRS.param** from LRS.Setup($\boldsymbol{\lambda}$). All outputs are constructed with input $\boldsymbol{\lambda}$. Collectively these elements are labeled **param**.

This algorithm can only be run once by the authority at the beginning of the system.

AddVoter On input **param**, AddVoter returns a new public-secret key pair generated using the (LRS.Gen). The public key is then added to **VL**.

This algorithm can be run once each by arbitrarily many eligible voters. The exact method of voter registration (and eligibility enforcement) is, like most voting systems, left largely open.

Run On input **vl** $\in$ **VL**, outputs an election-id $\mathcal{EI} \in \mathbb{N}$, Ring vl as the **elec-param**.

This algorithm can only be run once at the beginning of an election.

Vote On input $\boldsymbol{sk_i}, \boldsymbol{v}, \mathcal{EI}$, return ballot $\boldsymbol{b}$ as:

$LRS.Sign(\boldsymbol{sk_i}, \mathcal{EI}, \boldsymbol{vl}, E(\boldsymbol{pk_A}, \boldsymbol{v}))$.

[6] One option with VOTOR is not to encrypt the vote. This potentially makes it easier for the voter to verify their vote was received as they wished. However, it significantly increases trust on the anonymous channel used which no longer needs to collaborate with election authority to break privacy.

BB Adds $\boldsymbol{b}$ to **bb** and returns **accept**, if $LRS.Verify(\boldsymbol{b})$ and $\boldsymbol{b}$ is independent of all $b \in$ **bb**. Otherwise return **reject**.

Revoke Removes $\boldsymbol{b}$ from **bb** and returns **accept**, if $LRS.Verify(\boldsymbol{r})$ and $LRS.Link(\boldsymbol{r}, \boldsymbol{b})$. Otherwise return **reject**.

Tally For each $\boldsymbol{b}$ in $\boldsymbol{bb}, \boldsymbol{v} = D(\boldsymbol{sk_A}, \boldsymbol{b})$. Then tally the votes $\boldsymbol{v}$ according to any desired method.

4.3.2 Vote Submission

To construct a vote submission, the voter first signs their vote. The signed vote is then encrypted under the election tellers' public key. The result is then signed again, though under a different tag of the same election. Though we suggest the vote be constructed as described above ($S(E(S(v)))$), another alternative is $S(E(v))$—without the inner signature—provided *verifiable* mixnet is used as the second anonymous channel. This has no privacy implications and is purely an efficiency decision.

$S(E(S(v)))$ or $S(E(v))$ is revealed on the WBB in a timely manner, which places an increased onus on the anonymous channel used for submissions. Since the anonymous channel is only required to provide privacy against the election tellers, and the election authority/teller is the one receiving the submission, the fact that the WBB publishes encrypted votes upon receipt provides no additional information to the relevant adversary.

4.3.3 Mixing and Tallying

Once the voting period has ended, all votes on the web bulletin board BB which are valid and independent are passed through the election teller controlled anonymous channel. The result is then tallied to provide the election result.

4.3.4 Revocation

If at any point during the election a voter wishes to revoke their vote, they can upload a new signature on the same vote. It is possible to consider more complex revocation policies, however they do not change the fundamentals of the scheme.

4.3.5 Re-use

Once voting is over, or alternatively after voting, the voter should update their voting keys to prevent future key leakage. This is done according to either the trivial method of generating a new key and authenticating it with a signature of the old, or by the method outlined in an inherently forward secure linkable ring signature scheme. We present the first example of the latter we present in Chapter 5.

4.4 Security Assumptions

Here we will detail the security assumptions on which our scheme relies.

4.4.1 Anonymous Channels

We follow in our security definition of an anonymous channel from [GGOR14] who "formally define a secure, many-to-one anonymous channel which allows n players to simultaneously transmit messages to any single player P. Let X denote the multiset of honest senders' messages and Y denote the multiset output by P. Then the following conditions are satisfied in the presence of an adversary A:

Anonymity: Even if P is corrupt, A learns no more than X (and in particular gains no information on which messages came from which honest players).

Privacy: If P is honest, then A learns no information on X.

Reliability: If P is honest, then $X \subseteq Y$.

Non-Malleability: If P is honest, then $|Y| \leq n$ and the probability distribution of $Y \setminus X$ is independent from X."

For our purpose P and A are equivalent, and therefore the anonymity definition is of greatest interest. We make use of the linkable ring signatures to ensure the reliability of the overall system through an unreliable channel. Non-malleability is harder to ensure, and we will later discuss various mitigations to attacks on this property.

4.4.2 Forward Secure Linkable Ring Signatures

A *forward secure linkable ring signature* (FS-LRS) scheme is a tuple of seven algorithms (Setup, KeyGen, Sign, Verify, Link, PubKeyUpd, and PriKeyUpd).

- **param** $\leftarrow$ **Setup**(λ) is a randomised algorithm which on input a security parameter λ returns public parameters **param**.
- $(sk_i, pk_i) \leftarrow$**KeyGen(param)** is an algorithm which on input **param** returns a private/public key pair (sk_i, pk_i).
- $\sigma \leftarrow$ **Sign**$(event, n, \vec{pk}_t, sk, M, t)$ which, on input an event-id $event$, a group size n, a set $\vec{pk}_t$ of n public keys, a private key sk whose corresponding public key is in $\vec{pk}_t$, a message M and a time t, produces a signature σ.
- **accept**|**reject** $\leftarrow$ **Verify**$(event, n, \vec{pk}_t, M, \sigma, t)$ which, on input an event-id $event$, a group size n, a set $\vec{pk}_t$ of n public keys, a message-signature pair (M, σ), and time t, returns **accept** or **reject**. We define a signature σ as valid for $(event, n, \vec{pk}_t, M, t)$ if **Verify** outputs **accept**.
- **linked**|**unlinked** $\leftarrow$ **Link**$(event, t, n_1, n_2, \vec{pk}_{t_1}, \vec{pk}_{t_2}, M_1, M_2, \sigma_1, \sigma_2)$ which, on input an event-id $event$, time t, two group sizes n_1, n_2, two sets $\vec{pk}_{t_1}, \vec{pk}_{t_2}$ of n_1, n_2 public keys respectively, and two valid signature and message pairs $(M_1, \sigma_1, M_2, \sigma_2)$, outputs **linked** or **unlinked**.
- $Z_{t+1} \leftarrow$ **PubKeyUpd**(Z_t) which, on input a public key, Z at time t, produces a public key for time $t + 1$.
- $sk_{t+1} \leftarrow$ **PriKeyUpd**(sk_t) which, on input a private key sk at time t produces the corresponding private key for time $t + 1$.

Correctness. **FS-LRS** must satisfy the following:

- *Verification correctness:* Signatures signed correctly will verify.
- *Linking correctness:* Two honestly created signatures on the same event and time period will link if and only if they have the same signer.
- *Updating correctness:* For any time period within the life of the system, the secret key derived from the private key update function will create a valid signature on a ring, including the public key derived using the public key update function from its original public key.

4.4.3 Notions of Security

Security of FS-LRS has five aspects: unforgeability, anonymity, linkability, non-slanderability, and forward security. We take the following oracles as modelling the ability of the adversary to break the scheme:

- $pk_{i,t} \leftarrow \mathcal{JO}(t)$. The *Joining Oracle*, upon request, adds a new user to the system, and returns the public key pk of the new user at the current time t.
- $sk_{i,t} \leftarrow \mathcal{CO}(pk_i, t)$. The *Corruption Oracle*, on input a previously joined public key pk_i, returns the corresponding secret key sk_i at the current time t.
- $\sigma' \leftarrow \mathcal{SO}(event, n, \vec{pk}_t, pk_\pi, M, t)$. The *Signing Oracle*, on input an event-id $event$, a group size n, a set $\vec{pk}_t$ of n public keys, the public key of the signer pk_π, a message M, and a time t, returns a valid signature σ'.

In the sequel we sometimes omit the time and/or user subscripts t, i when those are clear from context. In particular, the public key in our scheme does not undergo updating, so pk_t will be independent of t. We also assume a random oracle which models the hash function.

- $h \leftarrow \mathcal{H}(x)$. The *Random Oracle*, on input x returns h independently at random. If x is repeated h will be returned again.

Unforgeability. The unforgeability of FS-LRS schemes is defined in the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$, in which $\mathcal{A}$ is given access to oracles $\mathcal{JO}, \mathcal{CO}, \mathcal{SO}$ and the random oracle:

1. $\mathcal{C}$ generates and gives $\mathcal{A}$ the system parameters **param**.

2. $\mathcal{A}$ may query the oracles according to any adaptive strategy, polynomially many times in the security parameter.

3. $\mathcal{A}$ gives $\mathcal{C}$ an event-id $event$, a group size n, a set $\vec{pk}_t$ of n public keys, a message M, a time t, and a signature σ.

$\mathcal{A}$ wins the game if:

i. **Verify**$(event, n, \vec{pk}_t, M, \sigma, t) =$ **accept**;

ii. all of the public keys in $\vec{pk}_t$ are query outputs of $\mathcal{JO}$;

iii. no public keys in $\vec{pk}_t$ have been input to $\mathcal{CO}$; and

iv. σ is not a query output of $\mathcal{SO}$.

We denote by

$$\mathbf{Adv}_{\mathcal{A}}^{Unf}(\lambda) = \Pr[\mathcal{A} \text{ wins the game}].$$

Definition 4. ***Unforgeability***. *An **LRS** scheme is unforgeable if for all PPT adversaries $\mathcal{A}$, **Adv**$_{\mathcal{A}}^{Unf}(\lambda)$ is negligible.*

Unconditional Anonymity. It should not be possible for an adversary $\mathcal{A}$ to tell the public key of the signer with a probability larger than $1/n$, where n is the cardinality of the ring, even assuming that the adversary has unlimited computing resources.

Specifically, unconditional anonymity for **FS-LRS** schemes is defined in the following game between a challenger $\mathcal{C}$ and an unbounded adversary $\mathcal{A}$ in which $\mathcal{A}$ is given access to oracle $\mathcal{JO}$:

1. $\mathcal{C}$ generates and gives $\mathcal{A}$ the system parameters **param**.

2. $\mathcal{A}$ may query $\mathcal{JO}$ according to any adaptive strategy.

3. $\mathcal{A}$ gives $\mathcal{C}$ an event-id e, a time t, a group size n, a set of $\vec{pk}_t$ of n public keys such that all of the public keys in $\vec{pk}_t$ are query outputs of $\mathcal{JO}$, a message M, and a time t. Parse the set $\vec{pk}_t$ as $\{pk_1, ..., pk_n\}$. $\mathcal{C}$ randomly picks $\pi \in \{1, ..., n\}$ and computes $\sigma_\pi =$

Sign$(e, n, \vec{pk}_t, sk_\pi, M, t)$, where sk_π is a valid private key corresponding to pk_π at time t. The signature σ_π is given to $\mathcal{A}$.

4. $\mathcal{A}$ outputs a guess $\pi' \in \{1, ..., n\}$.

We denote by

$$\mathbf{Adv}_{\mathcal{A}}^{Anon}(\lambda) = |Pr[\pi = \pi'] - \tfrac{1}{n}|.$$

Definition 5. ***Unconditional Anonymity***. *An **FS-LRS** scheme is unconditionally anonymous if for any unbounded adversaries $\mathcal{A}$, **Adv**$_{\mathcal{A}}^{Anon}(\lambda)$ is zero.*

Linkability. Linkability for **FS-LRS** is mandatory. It should be infeasible for a signer to generate two signatures such that they are determined to be **unlinked**.
Linkability for the **FS-LRS** scheme is defined in the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ in which $\mathcal{A}$ is given access to oracles $\mathcal{JO}, \mathcal{CO}, \mathcal{SO}$ and the random oracle:

1. $\mathcal{C}$ generates and gives $\mathcal{A}$ the system parameters **param**.
2. $\mathcal{A}$ may query the oracles according to any adaptive strategy.
3. $\mathcal{A}$ gives $\mathcal{C}$ an event-id e, time t, group sizes n_1, n_2, sets $\vec{pk}_{t_1}$ and $\vec{pk}_{t_2}$ of public keys of size n_1 and n_2, respectively, messages M_1, M_2, and signatures σ_1, σ_2.

$\mathcal{A}$ wins the game if

i. All of the public keys in $\vec{pk}_t$ are query outputs of $\mathcal{JO}$;

ii. **Verify**$(event, n_i, \vec{pk}_{t_i}, M_i, \sigma_i$,t) = **accept** for $i = 1, 2$ such that σ_i are not outputs of $\mathcal{SO}$;

iii. Fewer than two queries have been made to $\mathcal{CO}$ (hence, $\mathcal{A}$ can only have at most one private key); and

iv. **Link**(σ_1, σ_2) = $unlinked$.

We denote by

$$\mathbf{Adv}_{\mathcal{A}}^{Link}(\lambda) = Pr[\mathcal{A} \text{ wins the game}].$$

Definition 6. ***Linkability***. *An **FS-LRS** scheme is linkable if for all PPT adversaries $\mathcal{A}$, **Adv**$_{\mathcal{A}}^{Link}(\lambda)$ is negligible.*

Non-slanderability. Non-slanderability ensures that no signer can generate a signature which is determined to be **linked** with another signature which is not generated by the signer.
Non-slanderabilty for **FS-LRS** schemes is defined in the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$, in which $\mathcal{A}$ is given access to the oracles $\mathcal{JO}, \mathcal{CO}, \mathcal{SO}$ and the random oracle:

1. $\mathcal{C}$ generates and gives $\mathcal{A}$ the system parameters **param**.
2. $\mathcal{A}$ may query the oracles according to any adaptive strategy.
3. $\mathcal{A}$ gives $\mathcal{C}$ an event $event$, time t, group size n, a message M, a set of n public keys $\vec{pk}_t$, the public key of an insider $pk_\pi \in \vec{pk}_t$ such that pk_π has not been queried to $\mathcal{CO}$ or has not been included as the insider public key of any query to $\mathcal{SO}$. $\mathcal{C}$ uses the private key sk_π correspoing to pk_π to run **Sign**$(event, n, \vec{pk}_t, sk_\pi, M, t)$ and to produce a signature σ' given to $\mathcal{A}$.
4. $\mathcal{A}$ queries oracles adaptively, except that pk_π cannot be queried to $\mathcal{CO}$, or included as the insider public key of any query to $\mathcal{SO}$. In particular, $\mathcal{A}$ is allowed to query any public key which is not pk_π to $\mathcal{CO}$.
5. $\mathcal{A}$ delivers group size n^*, a set of n^* public keys $\vec{pk}_t^*$, a message M^*, and a signature $\sigma^* \neq \sigma'$.

A wins the game if

- **Verify**$(event, n^*, \vec{pk}_t^*, M^*, \sigma^*$, t) = **accept**;
- σ^* is not an output of $\mathcal{SO}$;
- all of the public keys in $\vec{pk}_t^*, \vec{pk}_t$ are query outputs of $\mathcal{JO}$;
- pk_π has not been queried to $\mathcal{CO}$; and
- **Link**(σ^*, σ') = **linked**.

We denote by

$$\mathbf{Adv}_{\mathcal{A}}^{NS}(\lambda) = Pr[\mathcal{A} \text{ wins the game}].$$

Definition 7. ***Non-slanderabilty***. *An **FS-LRS** scheme is non-slanderable if for any PPT adversaries $\mathcal{A}$, **Adv**$_{\mathcal{A}}^{NS}(\lambda)$ is negligible.*

Forward Security. Forward security ensures that it should not be possible for an adversary with a private key for a time period greater than t to create valid signatures for any time period less than or equal to t.

Forward Security is defined in the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ in which $\mathcal{A}$ is given access to oracles $\mathcal{JO}, \mathcal{CO}, \mathcal{SO}$ and the random oracle:

1. $\mathcal{C}$ generates and gives $\mathcal{A}$ the system parameters **param**.
2. $\mathcal{A}$ may query the oracles according to any adaptive strategy.
3. $\mathcal{A}$ gives $\mathcal{C}$ an event-id e, a group size n, a set $\vec{pk}_t$ of n public keys, a message M, a time t and a signature σ.

A wins the game if

i. **Verify**$(e, n, \vec{pk}_t, M, \sigma, t) =$ **accept**;

ii. all of the public keys in $\vec{pk}_t$ are query outputs of $\mathcal{JO}$;

iii. no public keys in $\vec{pk}_t$ have been input to $\mathcal{CO}$ at time t or earlier; and

iv. σ is not a query output of $\mathcal{SO}$.

We denote by

$$\mathbf{Adv}_{\mathcal{A}}^{FS}(\lambda) = Pr[\mathcal{A} \text{ wins the game}].$$

Definition 8. ***Forward Security***. *An **FS-LRS** scheme is forward secure if for any PPT adversaries $\mathcal{A}$, **Adv**$_{\mathcal{A}}^{FS}(\lambda)$ is negligible.*

4.5 VOTOR Analysis

In this section we will present reductions for the privacy and integrity of VOTOR to the underlying linkable/traceable ring signature scheme. We will also discuss wider issues in integrity, privacy, receipt freeness, revocation, and practicality.

4.5.1 Integrity

The integrity of the scheme is ensured by end-to-end verification of the following three key requirements: 'Cast as Intended', 'Collected as Cast', and 'Counted as Collected'. VOTOR, along with Helios and others, does not provide full Cast-As-Intended verification. We will discuss various countermeasures in our later discussion on integrity, but none provide an altogether desirable solution. Due to the use of signatures in our scheme, both the inputs to the finally tally and the contents of the WBB are signed by the voters. So for any attack on the integrity of this scheme, an adversary needs to produce valid message-signature pairs. This can be done in two ways.

One option is for the adversary to try to directly attack the linkable/traceable ring signature used. Since (some of) these signatures have a proof of security in the standard model, this seems likely to be an unfruitful endeavour. We will present the reduction from verifiability in VOTOR to the unforgeability of the underlying signature scheme in the next subsection.

The other option is to subvert the PKI. This can be achieved by attacking the preparation of the ledger to add public keys (whose corresponding private keys are known to the attacker), through adding additional voters, or by replacing the public keys of existing voters. The keys of honest users can also be effectively subverted by recovering the secret keys of voters at a later time, typically by means of a corrupt user-side client, or by altering the view of reality provided to the human voter, as we explain later in our discussion.

Individual Verifiability (Collected As Cast) v. Election Tellers and Authority

We will now sketch that breaking Individual Verifiability (Collected As Cast) v. Election Tellers and Authority breaks unforgeability in the underlying linkable ring signature. Hence, if the underlying ring signature is secure then attacks on Individual Verifiability (Collected As Cast) occur only when the underlying linkable ring signature key is exposed.

The adversary wishes to generate a valid ballot b_i for a voter i, who is not corrupted.

Let $\mathcal{C}$ be the linkable ring signature unforgeability challenger. We can construct a successful adversary $\mathcal{S}$ by simulating a challenger to a successful VOTOR individual verifiability (as collected) adversary $\mathcal{A}$.

1. $\mathcal{C}$ sends $\mathcal{S}$ **param**. $\mathcal{S}$ creates an empty voter list **VL**, a vote space m and authority keys pk_A, sk_A. $\mathcal{S}$ sends $\mathcal{A}$ **param**.

2. $\mathcal{A}$ may query any $\mathcal{JO}, \mathcal{CO}$ according to any adaptive strategy. $\mathcal{S}$ answers $\mathcal{JO}$ by calling $\mathcal{C}.\mathcal{JO}$ and adding the result to **VL**. Queries to $\mathcal{CO}$ are answered by sending these queries to $\mathcal{C}.\mathcal{CO}$ and returning its response.

3. $\mathcal{A}$ gives **vl**$\in$**VL** to $\mathcal{S}$. $\mathcal{S}$ calls **Run(VL)** and sends $\mathcal{A}(\mathcal{EI}, \textbf{vl}, \textbf{bb}, \textbf{election-param})$.

4. $\mathcal{A}$ may query $\mathcal{VO}, \mathcal{CO}$ according to any adaptive strategy. $\mathcal{S}$ answers $\mathcal{VO}$ by calling $\mathcal{C}.\mathcal{SO}$ and returning the result. Queries to $\mathcal{CO}$ are answered by sending these queries to $\mathcal{C}.\mathcal{CO}$ and returning its response.

5. $\mathcal{A}$ returns b.

6. $\mathcal{S}$ gives to $\mathcal{A}$ $(\mathcal{EI}, |n|, \textbf{vl}, b)$.

It now suffices to note that the definition of winning in the two games is nearly identical. In the $IndColl$ the only way to add to voter list vl is through the JO which implicitly satisfies criterion *ii* of Unforgabilty of LRS.[7]

$$\text{So } \mathbf{Adv}_{\mathcal{S}}^{Unf}(\lambda) = \mathbf{Adv}_{\mathcal{A}}^{IndColl}(\lambda).$$

Discussion

Having shown that an attack requires knowledge of the key, provided the underlying signature scheme is unforgeable, we now discuss mitigation techniques to key exposure.

Key Exposure The primary vulnerability of voter secret keys lies in the user-side computational devices that hold them. Corrupting the device may expose the key. While this style of attack cannot easily be prevented without a trusted signing device, it can be mitigated by sending a signed revocation message to the WBB, provided that the voter has retained a copy of their private key or pre-generated the revocation message, just in case.

[7] In $IndColl$ the ballot b must be unlinked to the $\mathcal{CO}$ queries, whereas in Unf, no keys in the ring are allowed to have been corrupted. The intuition for why this doesn't matter for security is as follows. If the adversary is able to use one of corrupted keys in the ring to create a ballot that doesn't link, they have broken linkability. Since this is not possible, the ballot b must be from one of the uncorrupted parties.

Corrupt Device Before analysing the scheme in detail we note that none of the suggested cryptographic measures are readily verified by humans. This can be overcome to some extent by using multiple devices and assuming at least one acts honesty, however this comes at a high useability cost. A simple, at least in theory, solution is to use a (human applied) blinding factor—code voting—which prevents the device from understanding, and meaningfully modifying, the vote which it submits. This is (abstractly) what happens with the codes in Scantegrity and Remotegrity, however the solution applied there requires the election tellers to know which voter they are interacting with to remove the blinding and send the confirmation codes. Such an approach is fundamentally at odds with the voter indistinguishability which VOTOR aims for. To achieve the desired properties in this way, we would require a human computable piece of asymmetric cryptography. While such systems are beginning to appear, we would require encryption to be efficiently computable—the current scheme has human computable decryption.

Vote Substitution Attack and Prevention A more complicated attack which an adversary might try to achieve is to replace the intended vote with another one, while making the voter believe that his own vote has duly appeared on the WBB. The idea is to tell each of multiple voters (who wish to vote the same way) that a certain entry on the WBB is their vote. Since the signatures are anonymous, the adversary hopes that each voter will believe them. This allows the adversary to replace all but one of the votes in the targeted group, even though all the attacked voters will believe that their vote was accurately recorded. This attack is predicated on adversarial control of the client device's user interface, even though the private key itself might remain inviolate in a trusted hardware module.

To defeat this attack, a voter can use the signatures' traceability to check that a purported vote posted on the WBB is positively theirs. To do so, the voter goes to another device, constructs a second signature for his original intended vote, and, without uploading it to the WBB, checks that the WBB vote traces as "linked" to the one just made. (Since the second signature is made on a message identical to the original, no harm will result should it be accidentally uploaded to the WBB, unless anonymous revocation is enabled.) If the two signatures are found to be "independent", then the voter knows that they are being played.

In a variant of VOTOR which uploads vote to the WBB in the clear, the voter can determine what vote the corrupt device did upload. This is done by comparing the new signature with all entries on the WBB. One signature on the WBB will have been formed using the voter's public

key and will thus be linked to the test signature. Since the vote is in plaintext, this reveals how the corrupt device voted. The test signature can then be released to revoke the forgery. However, the variant we mention here presents issues of fairness, since partial results are released.

Channel Malleability We have argued that due to the security of the signature the reliability of the channel between the user and the WBB is secure. However, the non-malleability of the channel is more difficult. A non-malleable channel would prevent the authority from selectively 'not receiving' certain ballots. If the channel had compulsory anonymity and the messages leaked no information this would be self-evidently true since the WBB would have nothing to base its selection on. However, neither of these assumptions are well founded; indeed in most cases they are outright false. One proposal is that various sub-bulletin boards are maintained by the competing parties in the election. All valid ballots appearing on these boards by a certain deadline must also appear on the WBB or the WBB is considered to be misbehaving. These sub-bulletin boards could post hashes of their contents into a major blockchain periodically as a form of time stamping. The validity of the contents of these boards is universally verifable.

Securely using Untrusted Devices with Limited Network Access The vote substitution attack detailed above requires the device to collude with several other devices. If the device does not have an open network connection, it cannot perform that attack, and hence the device assisting the voter does not threaten integrity.

This would require the construction and distribution of devices which are able to generate signatures but do not have an open network connection. The voter can then use these devices to generate their signature on their vote. The practical issue here is the size of the ring which is far too large to be transferred by QR code. The signed vote on the other hand would be transferable by QR or other visual medium. In summary, if the ring, and only the ring, can transferred to a device, this integrity attack with insecure devices would be resolved.[8]

The device could also construct an invalid signature. However, the voter will detect this when uploading the vote to the WBB or when running the verification algorithm on another device. Once the voter is satisfied with the signed ballot, they need to transfer it to a machine with a network connection that can then submit it. In variants of VOTOR where the ballot is sent unencrypted this could suffice to provide integrity against the assisting device. However, if

[8]Though not wider integrity issues and privacy.

the device also encrypts the ballot then it can encrypt a ballot other than that requested, an issue shared with Helios about which much has been written but no particularly desirable solution proposed.

Mandatorily Trusted Devices It is within the bounds of imagination that a government would issue a trusted device to each voter which enables them to vote. This would eliminate the threat posed by general malware but would place trust on the government. Since we are trying to avoid strong trust on the authority with VOTOR this is not a real option.

4.5.2 Privacy

The privacy of VOTOR relies on two assumptions. First, that the linkable/traceable ring signature does provide anonymity. The second and more interesting assumption is that Tor provides an anonymous channel. Below we will provide a reduction, assuming an anonymous channel, from voter indistinguishability to anonymity of the ring signature.

Voter Indistinguishability We show that our construction satisfies voter indistinguishability. That is, the adversary cannot identify a voter from their ballot. Since all the ballots are submitted over an anonymous channel, the foregoing suffices to ensure privacy.

Let $\mathcal{C}$ be the linkable ring signature anonymity challenger. We can construct a successful signature anonymity adversary $\mathcal{S}$, by simulating a challenger to a successful election voter indistinguishability adversary $\mathcal{A}$.

1. $\mathcal{C}$ sends $\mathcal{S}$ **param**. $\mathcal{S}$ creates an empty voter list **VL**, a vote space m, and authority keys pk_A, sk_A. $\mathcal{D}$ sends $\mathcal{A}$ **param**.

2. $\mathcal{A}$ may query any $\mathcal{JO}, \mathcal{CO}$ according to any adaptive strategy. S answers $\mathcal{JO}$ by calling $\mathcal{C}.\mathcal{JO}$ and adding the result to **VL**. Queries to $\mathcal{CO}$ are answered by sending these queries to $\mathcal{C}.\mathcal{CO}$ and returning its response.

3. $\mathcal{A}$ chooses **vl**$\in$**VL** and sends **vl** to $\mathcal{S}$. S calls **Run**(**vl**) and sends $\mathcal{A}$ ($\mathcal{EI}$, **vl**, **bb**, **election-param**).

4. $\mathcal{A}$ may query any $\mathcal{VO}, \mathcal{CO}$ according to any adaptive strategy. S answers $\mathcal{VO}$ by calling $\mathcal{C}.\mathcal{SO}$ and returning the result. Queries to $\mathcal{CO}$ are answered by sending these queries to $\mathcal{C}.\mathcal{CO}$ and returning its response.

5. $\mathcal{A}$ gives $\mathcal{S}$ $v \in m$, along with $\{b_1, ...b_i\}$ for the i corrupted voters and ouputs of the voting oracle.

6. $\mathcal{S}$ gives $\mathcal{C}$ an event-id $\mathcal{EI}$, a group size $|\mathbf{vl}|$, and a message v for which it receives σ_π. $\mathcal{S}$ now runs **BB** on all bs provided by $\mathcal{A}$. It runs **BB**(Vote(sk_π, v')) and **bb** is given to $\mathcal{A}$.

7. $\mathcal{A}$ outputs a guess $\pi' \in \{\mathbf{vl}\}$. $\mathcal{S}$ return π' to $\mathcal{C}$.

Therefore, we see that[9]

$$\mathbf{Adv}_{\mathcal{S}}^{Anon}(\lambda) = \mathbf{Adv}_{\mathcal{A}}^{VoterInd}(\lambda)$$

Discussion While recent leaked documents from the NSA suggest that Tor provides a somewhat secure anonymous channel, the particular way in which we use it provides two new avenues of attack. We weaken Tor by publishing votes on the WBB in a short time period after they are received. In addition, it is expected that many of the voters using Tor for an election will not use Tor normally. While it is unclear how this information would assist later relays, it provides significant additional information to the first relay.

For a summary of Tor, timing attacks and our countermeasures we refer the reader to Subsection 4.2.3 Anonymous Channel(s). If the first relay in the (Tor) network is corrupt, upon receiving an onion from an address it has not previously known, it assumes them to be a voter in the election. The corrupt relay then waits to see what votes appear on the WBB in the period soon after routing which will restrict the voter's privacy to a smaller set. If the relay correctly guesses that it was a vote that it passed on, the relay will know that the voter at the address it recorded voted one of a small subset of votes. This attack requires the corrupt Tor node to mount a traffic analysis and a timing attack. A similar attack can be mounted by any party able to observe the onion going to the first relay. However, this latter attack requires the additional assumption that the encrypted data was a Tor onion. Due to the large amount of traffic on Tor, widespread analysis of this kind could have a high cost per voter.

[9]Note that the unconditional anonymity only specifies access to $\mathcal{JO}$ however it assumes a computational unbounded adversary who can implicitly perform corruptions and signings. Definitions of standard anonymity for linkable ring signatures provide the oracle access we assume.

The voter can mitigate this attack by sending onions to other sites throughout the time period of the election. This forces the adversary to consider all votes published in related time intervals as possibly the voter's. Since the stronger version of this attack only works for the first Tor node, the voter should construct their circuit such that the party in whom they have the highest trust is first.

In addition, at the cost of inconveniencing voters, the WBB can delay the publishing of votes. This delay allows the WBB to publish larger sets at once thereby mitigating the timing attack. Unfortunately this option would require the voter to interact with the system on two separate occasions, once to vote and once to check that their vote was received. While this may be acceptable for certain voter groups, in general it would be expected to have a profoundly negative impact on the number of voters who actually check that their ballot was received.

What if the voter chooses a bad set of Tor nodes?

One notable form of vote selling the voter might try to engage in is to choose a set of Tor nodes which do not provide anonymity. For instance, by choosing only Tor nodes controlled by a certain government or organisation, the voter might try to reveal how they voted (at least down to their IP address). It is for this very reason that we recommend leaving a weak mixnet of election tellers in the scheme. Voters will normally achieve better privacy with VOTOR than Helios or Remotegrity, but even a voter wishing to de-anonymise themselves cannot get worse privacy.

4.5.3 Receipt freeness

The scheme does not provide coercion resistance under our strong definition. Indeed, we believe no remote scheme can do so.

In particular, the linkability of LRS can be used to provide proof to a coercer or vote buyer that the user did cast a certain vote: after the election closes, the voter issues a second signature that should trace with one of the votes published on the WBB. Alternatively, the vote seller can simply sell his private signing key.

Nevertheless, vote selling is less compelling and more difficult than in many other remote voting schemes, because the voter is able to revoke their vote, or even their voting credential, even after selling them. The vote buyer can always delay their payment until the election result

has been published and the seller's vote confirmed; however, such a payment delay is likely to make vote selling less appealing.

4.5.4 Revocation

Revocation of public keys is easy in our scheme. However, the difficulty of certifying new keys is vastly different between and in elections. Between elections, individual keys of voters can be revoked and re-certified without any effect on other voters. During an election, the ring of public keys—with respect to which voters are to create valid signatures—cannot be changed without invalidating all previously posted signatures.[10]

In general, most attack-mitigation approaches, be they to preserve integrity or receipt freeness, require voters (to be able) to revoke their key. However, revoking a voter's key during an election would disenfranchise that voter since we cannot certify new keys until the election has finished. One trivial option to resolve this is to allow disenfranchised voters to re-authenticate with new keys and revote if the election is close enough that the missed votes could have an influence. However, this is not ideal as it restricts the voter's privacy to the smaller set of disenfranchised voters. One may alternatively espouse the notion that users be held responsible for their keys, and treat a ballot as forfeit in the event that it needs to be revoked during the time that votes are being accepted.

Anonymous Revocation

The voter can anonymously revoke their vote by producing a new signature on the same message. Once submitted to the WBB the two pairs will return "linked" when run against the trace function.

It should be noted however, that some implementations of LRS or TRS allow construction of so called "dead" signatures. These signatures are created by an adversary and are of the form; given (m, σ) output (m, σ'), in other words the adversary is able to produce a new signature for any message signature pair he has previously seen. It is clear that anonymous revocation should not be allowed in any scheme based on TRS with "dead" signatures. The attack is trivial; the adversary, upon seeing a vote uploaded to the WBB that he wishes to cancel, will create a new signature on the same message and upload it, thus canceling the vote.

[10] There are alternatives which don't invalidate previous votes but they create substantial privacy issues.

Identity Revealing Revocation

If a traceable ring signature is used, identity revealing revocation can be achieved by uploading a signature on a new message. The trace on these two pairs will return "pk" the public key of the voter wishing to revoke their vote. This form of revocation has the added advantage that in the case of a close election result, the identities of the parties whose votes were not counted is known.

4.5.5 Practicality

The practical difficulties of implementing VOTOR vary greatly depending on the level of privacy, and to a lesser extent integrity, required. We present two analyses of VOTOR for different scenarios.

Helios Equivalence

The Helios voting system was initially pitched at low-coercion smaller elections. In these elections there is considerably less risk of specific malware being written and used to perform a vote substitution attack. The reduced number of voters would also make that attack harder since there are a smaller set of voters from which to find overlapping voters. The analysis for such a setting follows.

The pre-existence and extent of the Tor network removes a major implementation requirement for the election authority, effectively providing a high-traffic robust mixnet with thousands of pre-existing nodes, and growing. The only requirements to run an election using linkable or traceable ring signatures and Tor in our scheme, is a WBB web server, and an open source software client that implements the signature primitive for voters.

These requirements are very similar to the variants of Helios without multi-party mixnets. The security analysis comparing privacy in Section 4.6 motivates the use of VOTOR over Helios in these situations.

Better Robustness

In the more robust setting where multiple tellers are used, VOTOR has similar practical constraints to existing similar remote schemes using mixnets. The notable differences are that we do not require verifiable mixnets, significantly lowering the complexity of mixing.

Summary

Since most other remote voting schemes not only have similar conditions, but also additional requirements and constraints to achieve the same level of privacy, our VOTOR remote voting scheme appears to be significantly more practical and lightweight than other remote voting schemes with a similar level of security. Conversely, VOTOR offers better privacy than existing simple schemes in the literature today.

4.6 Comparison and Discussion

Having analysed the properties of a remote voting scheme based on linkable/traceable ring signatures, we compare and contrast this to two prominent remote voting schemes, Helios [Adi08] and Remotegrity [ZCC+13]. We also detail several attacks on JCJ/Civitas in support of our assertion that coercion resistance has not been achieved in the remote setting, these attacks exploit obvious issues in the assumptions and are well known.

Before doing this, we briefly note the advantage of our VOTOR framework over general mixnet based voting under the assumption that Tor is replaced with a traditional mixnet (which offers stronger guarantees, but under a more centralised trust assumption). Our scheme is indeed superior to a generic mixnet based construction. This follows because, if Tor is not acceptable for whatever reason, the voter can encrypt their signed vote under the threshold public key of the election tellers. They can then use an optional or compulsory mixnet for submission to achieve anonymity. The properties required of a mixnet for this purpose are weaker than those for general mixnet based voting. Specifically, we do not need to trust the correctness of the mixnet since the validity of the output is based on the ring signatures of the voters. However, the traditional flaws of mixnets remain: privacy breaks if all of its constituents collude, and each one of its constituents can perform denial of service attacks.

Table 4.1: Comparison Table of VOTOR, Helios and Remotegrity.

	VOTOR	Helios	Remotegrity
Practicalities			
Voter Authentication	Required	Required	Required
Revocation During Election	Partial	?	?
Revocation Between Elections	✓	✓	✓
Credential Persistence (Reuseability)	✓	?	✗
Confidentality of Votes and Voters			
Privacy			
v. Corrupt Election Tellers	✓	✗	✗
v. Corrupt Election Authority	✓	?	✗
v. Corrupt Device	✗	✗	✓
Vote-Selling Resistance	✓	✓	✗
Weak Receipt-Freeness	✓	✓	✓
Receipt-Freeness	✗	✗	✗
Coercion-Resistance	✗	✗	✗
Integrity			
Individual Verifiability (Cast)			
v. Corrupt Election Tellers	✓	✓	✓
v. Corrupt Election Authority	✓	✓	?
v. Corrupt Device	✗	✗	✓
Individual Verifiability (Collected)			
v. Corrupt Election Tellers	✓	✓	✓
v. Corrupt Election Authority	✓	✓	✓
v. Corrupt Device	✗	✗	✓
Universal Verifiability			
v. Corrupt Election Tellers	✓	✓	✓
v. Corrupt Election Authority	✓	✓	✓
v. Corrupt Device	✗	✗	✗

Table 1 summarises our findings comparing VOTOR with the two prominent online voting systems Helios and Remotegrity, using a rigorous analysis based on our revised and more realistic security model presented in this chapter.

4.6.1 JCJ/Civitas

The process of remote voter registration fundamentally fails in JCJ/Civitas to satisfy the robust definitions of receipt freeness and coercion resistance. In particular, JCJ/Civitas relies on a number of clearly false but necessary assumptions; we detail what happens when two of these are instantiated in the real world.

We will now describe Civitas in simplified terms. Civitas involves five kinds of agents: a supervisor, a registrar, voters, registration tellers, and tabulation tellers. It uses an underlying log service and digital signatures to ensure the security of the data used for verification. It also uses a trusted device to handle the mathematically complicated cryptography. After an initial setup, the registration tellers send the voter's private credential through the internet to the voter in a process that utilises the voter's trusted device. The voter can run a local algorithm to generate fake credentials which are indistinguishable to an adversary; the voter hands over a fake credential whenever they are coerced. The voter submits a vote using their real credential and finally the votes are tabulated.

"Trust Assumption 1" "An adversary cannot simulate a voter during registration" is unjustified in the remote setting; one might as well assume that the adversary cannot simulate the voter at the time of voting.[11]

"Trust Assumption 2a" "Each voter trusts at least one teller," is a direct contradiction with authority independence. We have already, and will continue to, argue that this is false for many real democracies, which tend to be dominated by a small number of powerful and colluding parties.

"Trust Assumption 2b" "and the channel from the voter to the voter's trusted registration

[11] There is a time escrow between registration and voting which might justify this, but would reduce Civitas from an election system to a time expander for an already secure election system.

teller is untappable." The existence of an untappable channel in the remote environment is a currently, and possible fundamentally, unsatisfiable requirement. Satisfying this requirement is therefore fundamentally at odds with the scheme's claim to be usable in a remote setting.

Attack 1 The coercer attends the voter right at the start of the election period and orders them to contact the tellers for their credentials then and there. The voter cannot signal the presence of the coercer to tellers without detection unless it already has a secret code, which becomes a circular argument. Since the credential is then displayed along with proofs the coercer learns the code. This attacks breaks trust assumption 2b that at least one channel is uncontrolled by the adversary. Since all channels include the same final display element (the voter's computer monitor) this assumption effectively resolves to a perfectly secure display, among other things.

Attack 2 The coercer removes the trusted device from the voter after registration but before voting. This breaks both trust assumption 1 and 2, however it is a perfectly valid way for the voter to transfer their vote.

Attack 3 The assumption of a cryptographically secure untappable channel has no known instantiation. The various types of deniable encryption-based systems either contain ephemeral data which must be destroyed, or would fail if certain adversarial provided ephemeral data was used. The attack here works by exploiting the failure of the channel used to provide the deniable properties required from an untappable channel.

We think that coercion resistance or a weaker property may yet be possible in remote voting, but the primitives required do not appear to exist.[12] While many "coercion resistant" remote voting schemes are of significant interest and provide valuable theoretical results, the form of abstract analysis which many schemes present fails to explain or justify the often unobtainable physical assumptions they are implicitly making.

[12]Even if humans could compute public key cryptography in their heads, without leaking side-channel information, this does not appear to be sufficient.

4.6.2 Helios

Helios was proposed by Ben Addida in "Helios: Web-based Open-Audit Voting" 2008. The extreme practicality of this scheme has led to significant adoption and additional literature [ADMP+09][BPW12][DVDGA12][TPLT13][CS13]. However, while Helios has been adopted most frequently in its practical form, we analyse its security in both practical and secure forms and compare to VOTOR.

Fundamental differences

A question which could be asked is, what makes VOTOR better than Helios ballots submitted through Tor? If encrypted ballot submission through Tor was done in isolation the process of checking ballot validity would break voter privacy. A mechanism is required to prevent over-voting and prove validity anonymously; this is precisely what linkable ring signatures provide in VOTOR.

Integrity

Helios and VOTOR have roughly similar integrity standards as detailed below.

Voter Authentication The problem of how to authenticate voters is highly related in both schemes. If the implementation of the schemes does not rely on pre-existing public keys authenticated to the voters, a voter impersonation attack is trivial for the election authority since it holds the credentials of each voter.

While in Helios voters are able to cast ballots as soon as they receive their credential, in a LRS based scheme all voters must first send back their public keys to allow the election authority to construct the voter ledger. Only once this is done can voters begin to sign and submit ballots. For all voters correctly recorded on the ledger, no further attacks on integrity are possible. Unlike Helios, in VOTOR a corrupt election authority, provided that it did not learn the corresponding secret keys, cannot insert votes for users who choose not to vote. This prevents a particularly dangerous attack in Helios since voters not voting are, intuitively, less likely to check.

However, it should be noted that the election authority could insert non-voting voters into the voter ledger in parallel to the Helios attack. However, a key difference is that such an

attack would have to be launched before ballot casting commenced in VOTOR rather than being possible at any point during ballot casting in Helios.

Ballot Substitution Another attack on Helios is ballot substitution where the election authority replaces the ballot of a voter who did vote with another ballot of the authority's choosing. This attack is not possible on LRS based schemes since the authority cannot forge signatures.[13] The removal of this attack has the additional advantage to the voter that if the ballot appearing is not what they intended, they know that their device is corrupt rather than knowing that either the election authority or the device is corrupt.

Incorrect Shuffling The issues of incorrect shuffling and decryption which are present in Helios have no parallel in linkable ring signature based schemes. This is easy to see since the ballot contains an unforgeable signature.

Bad Ballot Construction The issue of correct ballot construction is similar between Helios and VOTOR, with both schemes being dependent on the device assisting the voter. The voter can transfer and verify the audit data on another device in Helios, and we present multiple secure ballot construction methods for VOTOR.

Privacy

The initial specification of Helios provided no privacy against the single election authority used. Later implementations used a threshold of election authorities to distribute this trust. Nevertheless, the voter is still required to trust their privacy to a set of election authorities (tellers). Since the composition of this threshold is likely outside of the voters' control, a linkable ring signature based scheme using Tor, which provides personal control (in addition to the tellers), will likely be preferred by cautious and privacy-conscious voters.

[13] Signatures could also be used in Helios but this either creates privacy issues or converts the scheme into a version of VOTOR.

Receipt freeness

Neither scheme prevents coercion. Helios achieves weak receipt freeness if the randomness is properly disposed of. In a similar way, the updating of the secret key achieves this in VOTOR.

Vote Retraction

VOTOR allow voters to cancel their vote through double voting, restricting the way vote selling can be achieved. This has advantages over any implementation of Helios which does not allow votes to be retracted (if votes can be retracted in Helios, the forward receipt freeness property does not hold during the election, since the coercer could ask the voter to re-cast their vote).

Practicality

All values posted on the WBB are publicly verifiable and tally-able in VOTOR. Since this is the case, the extensive proofs and audits involving cut-and-choose in Helios are not required. In addition, no verifable mixnets are required which significantly reduces the cost of implementation.

4.6.3 Remotegrity

Remotegrity [ZCC+13] is an extension of Scantegrity II[CCC+08] which enables secure remote voting. The scheme is best known for preventing corrupt devices from learning the vote or changing the vote with non-negligible probability. Since the election authority cannot claim that a corrupt device sent incorrect codes, the election authority is unable to commit substitution attacks without incriminating itself.

Fundamental differences

Remotegrity and VOTOR make different decisions in pursuit of different goals. Remotegrity provides device independence by assigning each voter individual codes. VOTOR provides privacy from the election authority by making all voters (rather than their votes) indistinguishable. The individual codes of Remotegrity cannot be used in VOTOR because they are individual to the voters and hence reveal the corresponding voter's identity.

The Problem of Printing

The primary problem with Remotegrity is how to print the confirmation code sheets securely. Most current solutions reduce privacy to one device under the control of the election authority. While there have been proposed solutions [ECHA09b], the solutions have significant useability constraints, would need to be used very carefully to convince the voters and do not appear to have been Incorporated in practice. As we present our analysis, we will mention in passing the attacks possible if we assume the election authority remembers the codes, an assumption not unreasonable but not considered in the original paper.

Integrity

Remotegrity achieves integrity in regards to the computational device used by the voter. This follows since the device does not know the code which would lodge the vote it desires. The device implementing linkable ring signatures, on the other hand, knowing the voter's secret key, can sign any vote of its choosing. Remotegrity is therefore superior in terms of integrity against the computational device of the voter.

The integrity of Remotegrity against the election authority is based on the physical existence of a tamper-evident device. This device allows the voter to prove that they could not have known the value required to submit the vote the election authority claims they did. In linkable ring signature based schemes, the integrity against election authorities is based on standard cryptographic assumptions. We therefore consider linkable ring signature based schemes to have higher integrity in regards to the election authority.

Cut-and-Choose on One Ballot Scantegrity II uses a cut and choose protocol to convince voters that their ballots were cast as intended; that is, the confirmation codes refer to correct ballot selection. Remotegrity seems to propose to send a single ballot to each voter. It is unclear to us how the voter can be convinced that the confirmation codes—they submitted—accurately record their vote. The authors of Remotegrity have proposed a solution to this problem, but it remains unconvincing to some in the community.

Privacy

Remotegrity, like the Scantegrity system it is based on, offers no privacy against the election authority. Schemes based on linkable ring signatures do. This follows because the confirmation codes returned to the election authority were initially seen by its printers, the process of tallying also relies on the the existence of a privacy preserving teller.

Receipt freeness

Neither of these schemes provide resistance to coercion. However the nature of the tamper-evident device in Remotegrity allows a coercer or a vote buyer to achieve high assurance that the vote they receive has not been cast and cannot be cast by the original voter, once they have received the tamper-evident device. This contrasts with linkable ring signature schemes where the voter—on giving their secret key to a coercer or vote buyer—will likely still retain a copy, allowing them to double vote and thereby invalidating both votes.

Practicality

The Remotegrity system requires a code for every possible vote. This makes it impractical for expressive elections (ones with complex ballots). In contrast, linkable ring signatures can be used to sign any message and hence any type of vote.

4.6.4 Future work

One of the most significant issues with VOTOR is the reliance on the voter's computation device, however limited that reliance remains. While we have shown how to mitigate several attacks involving a corrupt or compromised user device, removing this class of attacks completely would be a major improvement. There seem to be two avenues of approach to this problem.

The first is to distribute or construct confirmation codes in the style of Remotegrity, and get the voter to submit these as their vote. As well as the difficulty of convincing the voter that the codes will correctly capture their vote, we would require the additional, unmet property that the codes could not be traced to the voter.

The second, "blue-sky", option is to create some form of partially human-bound or human-computable blinded linkable/traceable ring signature, depending on a secret that does not leave the human mind. Here, we might benefit from the observation that such a secret can be very short indeed, since any use of an incorrect code could result in a valid-looking signature whose presence the human will not expect. This ambitious approach would allow the voter to sign without reliance on a computational device or having to encrypt their vote before signing.

The version of this scheme appearing in the published paper was only secure against a computationally bounded adversary. This raises concerns about the long term privacy of votes. There were solutions for ring signatures which are called unconditionally anonymous, which offer privacy against computational unbounded adversaries. There has been some progress towards this recently with [LASZ14] providing proofs in the random oracle model. However, no such scheme at that time offered forward secrecy. The construction of a forward secure linkable ring signature scheme with unconditional anonymity would allow a stronger instantiation of VOTOR. Since the original version of the paper, we have developed such a signature scheme albeit under strong cryptographic assumptions, and its construction is detailed in the next chapter.

4.7 Conclusion

We have presented a practical remote voting scheme called VOTOR which provides security from the election authority and tellers. It uses linkable ring signatures and commodity anonymous channels, such as Tor. We provide an explanation of how to adapt Tor to provide higher levels of privacy for this setting. We present a framework for evaluating the security of voting schemes which we use to analyse VOTOR. The security of VOTOR compares favourably to popular and recent schemes such as Helios and Remotegrity on many aspects (but not all; see table 1), largely due to the simple reduction to secure cryptographic primitives. In addition, its simplicity and use of freely available existing systems enable high levels of practicality and voter verification which surpass the other systems.

Chapter 5

Forward Secure Linkable Ring Signatures

Historically, privacy was almost implicit, because it was hard to find and gather information. But in the digital world, whether it's digital cameras or satellites or just what you click on, we need to have more explicit rules - not just for governments but for private companies.

(Bill Gates)

Abstract

We present the first candidate strategy for a linkable ring signature scheme with both unconditional anonymity and a forward secure key update: which would be a powerful tool with a direct practical application—indeed our primary motivation—to satisfy the simultaneous operational requirements of internet voting. We propose a formal security model with multiple security properties, and offer a construction based on the hardness of finding discrete logarithms and (for forward security) multilinear maps of moderate degree. We deliver efficient security reductions which are incidentally much tighter than existing (non-forward secure) linkable ring signatures. We then discuses how our contribution elegantly solves a number of problems heretofore unsolved in the important application of (multi-election) practical internet voting. Unfortunately, all candidate multilinear maps are currently broken and hence this scheme is unrealisable until they are fixed.

5.1 Introduction

Ring signatures, and especially linkable ring signatures, garner much interest in the applied cryptographic community for their promise to simplify certain aspects of the notoriously hard problem of remote electronic voting, with its many and sometimes conflicting security requirements. In particular, linkability [LASZ14] or the closely related notion of traceability [FS07] make it easy to detect when the same signer has issued two or more signatures on a similar issue, thereby preventing double spending in an electronic cash system, or here, double voting in a single election.

However, so far these signatures have not assisted in simultaneously resolving two critical issues in electronic voting. These two issues are: (1) how to register voters, and; (2) how to ensure their long term privacy. To address these issues, offline key update lets the potentially costly registration of a voter's public key happen once, whereafter the corresponding private key can be refreshed or updated multiple times, efficiently and *non-interactively*, for use in future elections. In this context, forward security refers to the notion that the leakage or compromise of an updated private key will not compromise one's privacy in a past election—or more generally, let an attacker forge signatures ostensibly in the past. For practical electoral systems in particular, it is important that the public key update mechanism be efficient and non-interactive. The ideal public key update is the identity function, or "no-op." The private key update serves to provide forward security to protect old elections against future data exposure and compromises.

The related but different notion of unconditional anonymity refers to the inability, even by a computationally unbounded attacker, to identify a signer without knowledge of their private key. This notion is important to protect the voter against future increases in computational power (or cryptanalytic attacks, or quantum computers), once they have destroyed their private key after it is no longer needed. Together with linkability, these features make substantially easier the task of designing a secure and useable remote election protocol. Our forward secure linkable ring signature scheme, when dropped into a number of existing election protocols, directly results in a straightforward and secure electronic voting solution without the cumbersome, complicated and procedurally risky steps that would normally be necessary.

Unfortunately—as often with the contradictory requirements of voting—it is easy to convince oneself that without *context-aware* public key update (and thus especially with no public key update at all), anonymity can only hold unconditionally if no *authentic* private key for the relevant signing ring is ever leaked, not even after having been refreshed. Indeed, if an adversary knows a voter's *authentic* private key, he can *always* trivially deanonymise their current and future votes using the linkability feature. The same is true for past votes if a past private key can be recovered, by brute-force or by breaking a hardness assumption, given an authentic current key. In light of this, we deliberately choose to focus on the problem of achieving unconditional anonymity but only computational forward security under non-interactive key update. We leave it as an open problem to construct an *unconditionally forward secure* linkable ring signature with a context-aware private key update mechanism, e.g., that incorporates other ring members' public key information.

5.1.1 Our Results

We present the first candidate strategy for a linkable ring signature with unconditional anonymity and forward secure key update. This result would enable significantly more simple and secure remote electronic voting, even within the framework of existing electronic voting protocols, and opens the door to a number of simplified general anonymous authentication protocols for online systems.

To achieve our result we construct a linkable ring signature from unconditionally hiding commitments, and make sparing use of a multilinear map [GGH13, LSS14] to lift it to multiple "epochs". Without forward security or key update, our results are inspired by the linkable ring scheme of [LWW04], which—incidentally—we vastly improve via much tighter security reductions[1]. To get forward security, we appeal to an n-multilinear map as a tool from which we construct an n-time private key update mechanism with zero public key update. We prove security of the scheme, including forward security, based on information theoretic considerations and/or the Discrete Logarithm assumption and a new convenient multilinear assumption, which we prove equivalent to the classic multilinear Decoding Problem [GGH13].

[1]Our scheme is inspired from Liu et al. [LASZ14], who provided proofs which in the worst case were exponential in the number of users. Our proofs and reductions are nearly independent of the number of users, and the same techniques can directly apply to their construction. Like Liu et al, we make use of Pedersen commitments [Ped91] and the forking lemma [PS96a] in the random oracle model.

In supplement and in passing, we briefly mention a combinatorial construction that leverages k instances of our n-time basic construction to provide $\binom{n+2}{k+1}$ update periods. This is gained at the expense of requiring private keys of size $\binom{n}{k-1}$ and a strengthened hardness assumption no longer equivalent to the Decoding Problem assumption, for any fixed choice of $k \leq n$. The extended scheme might have uses in uncommon consultations requiring rapid-fire votes over an extended period of time, something that cannot be practically envisioned without remote voting. However, we consider our main scheme to be the core result of this chapter, as the most relevant to the current practice of electronic voting.

5.1.2 Related Work

Group signatures were presented by Chaum et al [CvH91]. They allow the members of a group to generate signatures which can only be verified as emanating from one signer within that group, with the additional property that the signature can be "opened" to reveal the signer. The ability to "open" a signature is an important requirement in certain managed applications, but presents an unacceptable privacy loophole in the context of electronic voting.[2]

Ring signatures are a variation or simplification of group signatures which do not allow "opening" of signatures, and hence, do not have those privacy issues. Ring signatures were first presented by Rivest et al [RST01b] as a way to leak secrets anonymously. Since then, many variants have been proposed to suit a large number of applications. For the application of voting, double voting becomes a major issue which vanilla ring signatures are not easily able to rectify. For this application, therefore, linkable ring signatures [LWW04] and traceable ring signatures [FS07] have been proposed. Neither of [LWW04, FS07] or their variants provide forward security; hence in a voting application they would require impractically frequent re-registration of new keys to ensure acceptable levels of privacy.

Subsequent results in that area include Liu et al. [LASZ14], who presented a linkable ring signature with unconditional anonymity, but still without forward security. Our scheme addresses this shortcoming, by providing an offline (non-interactive) key update mechanism with forward security (as well as much improved security reduction tightness) over the previous schemes.

[2] In the UK there is a requirement that a judge be able to order a voter's ballot revealed. In this case group signatures would be desirable.

Multilinear maps. Following the success of bilinear maps in cryptography, multilinear maps and their applications in relation to cryptography were studied at a theoretical level by Boneh and Silverberg [BS03]. Nearly a decade later Garg et al [GGH13] proposed the first practical candidate construction, based on lattice problems. There have since been several additional candidates from lattice and number-based assumptions, as well as attacks and repairs [CLT13, CHL$^+$15, GGH15, LSS14, BWZ14]. Our scheme relies on a generalisation of the Discrete Logarithm problem, which is a weaker assumption than the myriad of Diffie-Hellman variants typically found in cryptographic constructions based on bilinear or multilinear maps. However, it should be noted that there is no currently accepted candidate for multilinear maps and hence the construction in this work is currently unrealisable. Our tighter security reductions are still a valuable contribution for an unconditionally anonymous linkable ring signature scheme without forward security. We will discuss in subsection 5.3.2 the major issues at present in multilinear map constructions.

Voting systems. In the world of election systems research, the recent Helios [Adi08] protocol is, perhaps, the best known secure internet voting scheme. It has seen a significant variety of expansions and applications [TPLT13, DVDGA12], but one of its shortcomings is that the voters have to place (too) much trust on the election authority. Our linkable ring signature construction would fit nicely within the Helios protocol to enable powerful anonymous authentication and achieve privacy against the election authority, a property which is not achieved by most implementations of Helios[3]. Our construction also fits neatly into VOTOR, which we introduced in the previous chapter. More generally, and beyond election systems, our new signature scheme can be used as a general rate-limited[4] anonymous authentication system with information theoretic privacy and forward secrecy.

5.2 Security Definitions

A *forward secure linkable ring signature* (FS-LRS) scheme is a tuple of seven algorithms (Setup, KeyGen, Sign, Verify, Link, PubKeyUpd, and PriKeyUpd).

[3]According to the standard [Adi10], Helios uses a technique called a mixnet to distribute the power of the authority in respect to privacy. Even for Helios implementations that use this technique, gaining privacy as a function of authentication still provides strong additional privacy guarantees.

[4]Rate limitation in the context of authentication refers to an intentional bound on the number of uses, typically one, that can be made of a credential on a given target.

- **param** $\leftarrow$ **Setup**(λ) is a randomised algorithm which on input a security parameter λ returns public parameters **param**.

- $(sk_i, pk_i) \leftarrow$**KeyGen(param)** is an algorithm which on input **param** returns a private/public key pair (sk_i, pk_i).

- $\sigma \leftarrow$ **Sign**$(event, n, \vec{pk}_t, sk, M, t)$ which, on input an event-id $event$, a group size n, a set $\vec{pk}_t$ of n public keys, a private key sk whose corresponding public key is in $\vec{pk}_t$, a message M and a time t, produces a signature σ.

- **accept**|**reject** $\leftarrow$ **Verify**$(event, n, \vec{pk}_t, M, \sigma, t)$ which, on input an event-id $event$, a group size n, a set $\vec{pk}_t$ of n public keys, a message-signature pair (M, σ), and time t, returns **accept** or **reject**. We define a signature σ as valid for $(event, n, \vec{pk}_t, M, t)$ if **Verify** outputs **accept**.

- **linked**|**unlinked** $\leftarrow$ **Link**$(event, t, n_1, n_2, \vec{pk}_{t_1}, \vec{pk}_{t_2}, M_1, M_2, \sigma_1, \sigma_2)$ which, on input an event-id $event$, time t, two group sizes n_1, n_2, two sets $\vec{pk}_{t_1}, \vec{pk}_{t_2}$ of n_1, n_2 public keys respectively, and two valid signature and message pairs $(M_1, \sigma_1, M_2, \sigma_2)$, outputs **linked** or **unlinked**.

- $Z_{t+1} \leftarrow$ **PubKeyUpd**(Z_t) which, on input a public key, Z at time t, produces a public key for time $t + 1$.

- $sk_{t+1} \leftarrow$ **PriKeyUpd**(sk_t) which, on input a private key sk at time t produces the corresponding private key for time $t + 1$.

Correctness. **FS-LRS** must satisfy the following:

- *Verification correctness:* Signatures signed correctly will verify.

- *Linking correctness:* Two honestly created signatures on the same event and time period will link if and only if they have the same signer.

- *Updating correctness:* For any time period within the life of the system, the secret key derived from the private key update function will create a valid signature on a ring, including the public key derived using the public key update function from its original public key.

5.2.1 Notions of Security

Security of FS-LRS has five aspects: unforgeability, anonymity, linkability, non-slanderability, and forward security. We take the following oracles as modelling the ability of the adversary to break the scheme:

- $pk_{i,t} \leftarrow \mathcal{JO}(t)$. The *Joining Oracle*, upon request, adds a new user to the system, and returns the public key pk of the new user at the current time t.
- $sk_{i,t} \leftarrow \mathcal{CO}(pk_i, t)$. The *Corruption Oracle*, on input a previously joined public key pk_i, returns the corresponding secret key sk_i at the current time t.
- $\sigma' \leftarrow \mathcal{SO}(event, n, \vec{pk}_t, pk_\pi, M, t)$. The *Signing Oracle*, on input an event-id $event$, a group size n, a set $\vec{pk}_t$ of n public keys, the public key of the signer pk_π, a message M, and a time t, returns a valid signature σ'.

In the sequel we sometimes omit the time and/or user subscripts t, i when those are clear from context. In particular, the public key in our scheme does not undergo updating, so pk_t will be independent of t. We also assume a random oracle which models the hash function.

- $h \leftarrow \mathcal{H}(x)$. The *Random Oracle*, on input x returns h independently at random. If x is repeated h will be returned again.

Unforgeability. The unforgeability of FS-LRS schemes is defined in the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$, in which $\mathcal{A}$ is given access to oracles $\mathcal{JO}, \mathcal{CO}, \mathcal{SO}$ and the random oracle:

1. $\mathcal{C}$ generates and gives $\mathcal{A}$ the system parameters **param**.
2. $\mathcal{A}$ may query the oracles according to any adaptive strategy, polynomially many times in the security parameter.
3. $\mathcal{A}$ gives $\mathcal{C}$ an event-id $event$, a group size n, a set $\vec{pk}_t$ of n public keys, a message M, a time t, and a signature σ.

$\mathcal{A}$ wins the game if:

i. **Verify**$(event, n, \vec{pk}_t, M, \sigma, t) =$ **accept**;

ii. all of the public keys in $\vec{pk}_t$ are query outputs of $\mathcal{JO}$;

iii. no public keys in $\vec{pk}_t$ have been input to $\mathcal{CO}$; and

iv. σ is not a query output of $\mathcal{SO}$.

We denote by

$$\mathbf{Adv}_{\mathcal{A}}^{Unf}(\lambda) = \Pr[\mathcal{A} \text{ wins the game}].$$

Definition 9. ***Unforgeability***. *An **LRS** scheme is unforgeable if for all PPT adversaries $\mathcal{A}$, **Adv**$_{\mathcal{A}}^{Unf}(\lambda)$ is negligible.*

Unconditional Anonymity. It should not be possible for an adversary $\mathcal{A}$ to tell the public key of the signer with a probability larger than $1/n$, where n is the cardinality of the ring, even assuming that the adversary has unlimited computing resources.

Specifically, unconditional anonymity for **FS-LRS** schemes is defined in the following game between a challenger $\mathcal{C}$ and an unbounded adversary $\mathcal{A}$ in which $\mathcal{A}$ is given access to oracle $\mathcal{JO}$:

1. $\mathcal{C}$ generates and gives $\mathcal{A}$ the system parameters **param**.

2. $\mathcal{A}$ may query $\mathcal{JO}$ according to any adaptive strategy.

3. $\mathcal{A}$ gives $\mathcal{C}$ an event-id e, a time t, a group size n, a set of $\vec{pk}_t$ of n public keys such that all of the public keys in $\vec{pk}_t$ are query outputs of $\mathcal{JO}$, a message M, and a time t. Parse the set $\vec{pk}_t$ as $\{pk_1, ..., pk_n\}$. $\mathcal{C}$ randomly picks $\pi \in \{1, ..., n\}$ and computes $\sigma_\pi =$ **Sign**$(e, n, \vec{pk}_t, sk_\pi, M, t)$, where sk_π is a valid private key corresponding to pk_π at time t. The signature σ_π is given to $\mathcal{A}$.

4. $\mathcal{A}$ outputs a guess $\pi' \in \{1, ..., n\}$.

We denote by

$$\mathbf{Adv}_{\mathcal{A}}^{Anon}(\lambda) = |Pr[\pi = \pi'] - \tfrac{1}{n}|.$$

Definition 10. ***Unconditional Anonymity***. *An **FS-LRS** scheme is unconditionally anonymous if for any unbounded adversaries $\mathcal{A}$, **Adv**$_{\mathcal{A}}^{Anon}(\lambda)$ is zero.*

Linkability. Linkability for **FS-LRS** is mandatory. It should be infeasible for a signer to generate two signatures such that they are determined to be **unlinked**.
Linkability for the **FS-LRS** scheme is defined in the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ in which $\mathcal{A}$ is given access to oracles $\mathcal{JO}, \mathcal{CO}, \mathcal{SO}$ and the random oracle:

1. $\mathcal{C}$ generates and gives $\mathcal{A}$ the system parameters **param**.
2. $\mathcal{A}$ may query the oracles according to any adaptive strategy.
3. $\mathcal{A}$ gives $\mathcal{C}$ an event-id e, time t, group sizes n_1, n_2, sets $\vec{pk}_{t_1}$ and $\vec{pk}_{t_2}$ of public keys of size n_1 and n_2, respectively, messages M_1, M_2, and signatures σ_1, σ_2.

$\mathcal{A}$ wins the game if

i. All of the public keys in $\vec{pk}_t$ are query outputs of $\mathcal{JO}$;

ii. **Verify**$(event, n_i, \vec{pk}_{t_i}, M_i, \sigma_i$,t) = **accept** for $i = 1, 2$ such that σ_i are not outputs of $\mathcal{SO}$;

iii. Fewer than two queries have been made to $\mathcal{CO}$ (hence, $\mathcal{A}$ can only have at most one private key); and

iv. **Link**(σ_1, σ_2) = $unlinked$.

We denote by

$$\mathbf{Adv}_{\mathcal{A}}^{Link}(\lambda) = Pr[\mathcal{A} \text{ wins the game}].$$

Definition 11. ***Linkability***. *An **FS-LRS** scheme is linkable if for all PPT adversaries $\mathcal{A}$, **Adv**$_{\mathcal{A}}^{Link}(\lambda)$ is negligible.*

Non-slanderability. Non-slanderability ensures that no signer can generate a signature which is determined to be **linked** with another signature which is not generated by the signer.
Non-slanderabilty for **FS-LRS** schemes is defined in the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$, in which $\mathcal{A}$ is given access to the oracles $\mathcal{JO}, \mathcal{CO}, \mathcal{SO}$ and the random oracle:

1. $\mathcal{C}$ generates and gives $\mathcal{A}$ the system parameters **param**.

2. $\mathcal{A}$ may query the oracles according to any adaptive strategy.

3. $\mathcal{A}$ gives $\mathcal{C}$ an event $event$, time t, group size n, a message M, a set of n public keys $\vec{pk}_t$, the public key of an insider $pk_\pi \in \vec{pk}_t$ such that pk_π has not been queried to $\mathcal{CO}$ or has not been included as the insider public key of any query to $\mathcal{SO}$. $\mathcal{C}$ uses the private key sk_π correspoing to pk_π to run **Sign**$(event, n, \vec{pk}_t, sk_\pi, M, t)$ and to produce a signature σ' given to $\mathcal{A}$.

4. $\mathcal{A}$ queries oracles adaptively, except that pk_π cannot be queried to $\mathcal{CO}$, or included as the insider public key of any query to $\mathcal{SO}$. In particular, $\mathcal{A}$ is allowed to query any public key which is not pk_π to $\mathcal{CO}$.

5. $\mathcal{A}$ delivers group size n^*, a set of n^* public keys $\vec{pk}_t^*$, a message M^*, and a signature $\sigma^* \neq \sigma'$.

A wins the game if

- **Verify**$(event, n^*, \vec{pk}_t^*, M^*, \sigma^*$, t) = **accept**;
- σ^* is not an output of $\mathcal{SO}$;
- all of the public keys in $\vec{pk}_t^*, \vec{pk}_t$ are query outputs of $\mathcal{JO}$;
- pk_π has not been queried to $\mathcal{CO}$; and
- **Link**(σ^*, σ') = **linked**.

We denote by

$$\mathbf{Adv}_{\mathcal{A}}^{NS}(\lambda) = Pr[\mathcal{A} \text{ wins the game}].$$

Definition 12. ***Non-slanderabilty***. *An **FS-LRS** scheme is non-slanderable if for any PPT adversaries $\mathcal{A}$, **Adv**$_{\mathcal{A}}^{NS}(\lambda)$ is negligible.*

Forward Security. Forward security ensures that it should not be possible for an adversary with a private key for a time period greater than t to create valid signatures for any time period less than or equal to t.

Forward Security is defined in the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ in which $\mathcal{A}$ is given access to oracles $\mathcal{JO}, \mathcal{CO}, \mathcal{SO}$ and the random oracle:

1. $\mathcal{C}$ generates and gives $\mathcal{A}$ the system parameters **param**.
2. $\mathcal{A}$ may query the oracles according to any adaptive strategy.
3. $\mathcal{A}$ gives $\mathcal{C}$ an event-id e, a group size n, a set $\vec{pk}_t$ of n public keys, a message M, a time t and a signature σ.

A wins the game if

i. **Verify**$(e, n, \vec{pk}_t, M, \sigma, t) =$ **accept**;

ii. all of the public keys in $\vec{pk}_t$ are query outputs of $\mathcal{JO}$;

iii. no public keys in $\vec{pk}_t$ have been input to $\mathcal{CO}$ at time t or earlier; and

iv. σ is not a query output of $\mathcal{SO}$.

We denote by

$$\mathbf{Adv}_{\mathcal{A}}^{FS}(\lambda) = Pr[\mathcal{A} \text{ wins the game}].$$

Definition 13. ***Forward Security****. An* ***FS-LRS*** *scheme is forward secure if for any PPT adversaries* $\mathcal{A}$*,* $\boldsymbol{Adv}_{\mathcal{A}}^{FS}(\lambda)$ *is negligible.*

5.3 Multilinear Maps

Our notation is similar to that used by Zhandry in [Zha14].

Let $\mathcal{E}$ be an $l-$linear map over additive cyclic groups $[\mathbb{G}]_1, .., [\mathbb{G}]_l$ of prime order p, where $[\mathbb{G}]_0 = \mathbb{Z}_q$ and all $[\mathbb{G}]_i$ for $i = 1, ..., l$ are homomorphic to $(\mathbb{Z}_q, +)$. Let $[\alpha]_i$ denote the element $\alpha \in \mathbb{Z}_q$ raised to the level-i group $[\mathbb{G}]_i$, for $i \in (0, .., l)$. Let $\alpha \in_R [\mathbb{G}]_i$ denote the random sampling of an element in $[\mathbb{G}]_i$.

We have access to the following efficient functions:
The (cross-level multiplication) multilinear **map**, denoted $\mathcal{E}$, takes two elements $[\alpha]_i, [\beta]_j$ and returns $[\alpha * \beta]_{i+j}$.
The addition, **Add** or $+$, given two elements $[\alpha]_i, [\beta]_i$ returns $[\alpha + \beta]_i$.
The negation, **Neg** or $-$, given one element $[\alpha]_i$ returns $[-\alpha]_i$.

Since the level-0 group $[\mathbb{G}]_0$ is the field $\mathbb{Z}_q$, we can efficiently perform multiplications and inversions at level 0. The cryptographic security of multilinear maps requires that multiplication *within* any of the higher-level $[\mathbb{G}]_i$ be hard.

5.3.1 Multilinear Assumptions

For convenience, we will prove the security of our construction using the following hard problem, which we call Equivalent Decoding Problem, and which we prove below to be equivalent to the classic Decoding Problem from [GGH13]. We define:

Definition 14 ($(\kappa, h)-$Equivalent Decoding Problem ($(\kappa, h)-$EDP)). *For any probabilistic polynomial time algorithm $\mathcal{A}$, the probability $Pr[\mathcal{A}([\alpha]_0, [\alpha * x]_h, [x]_\kappa) = [x]_j]$ is negligible, where $j < \kappa$, $\alpha, x \in_R \mathbb{Z}_q$ and $h \geq \kappa$.*

We recall that the core problem in multilinear maps as stated in [GGH13] is:

Definition 15 (Multilinear Discrete-log Problem (MDLP)). *For any probabilistic polynomial time algorithm $\mathcal{A}$, the probability $Pr[\mathcal{A}([\alpha]_1) = [\alpha]_0]$ is negligible, where $\alpha \in_R \mathbb{Z}_q$.*

MDLP is a specific instance of the Generalised Decoding Problem [GGH13]:

Definition 16 (i-Decoding Problem $(i-DP)$). *For any probabilistic polynomial time algorithm $\mathcal{A}$, the probability $Pr[\mathcal{A}([\delta]_i) = [\delta]_j]$ is negligible, where $j < i$ and $\delta \in_R \mathbb{Z}_q$.*

Theorem 17. *(κ, h)-EDP is equivalent to i-DP, for $i = \kappa$:*

We prove the theorem by reduction in both directions:

Proof. Given an (κ, h)-EDP instance $([\alpha]_0, [\alpha * x]_h, [x]_\kappa)$ we simulate an i-DP instance $[\delta]_i$ as, $[\delta]_i = [x]_\kappa$. Having obtained the output from a successful adversary $\mathcal{A}$ to DP, $\mathcal{A}([\delta]_i) \rightarrow [\delta]_j$ for $j < i$, we return $[\delta]_j$ as the answer to the EDP instance.
Therefore the probability of success against EDP is identical to $\mathcal{A}$ against DP.

Conversely, given an i-DP instance $([\delta]_i)$ we simulate (κ, h)-DP instance $([\alpha]_0, [\alpha * x]_h, [x]_\kappa)$ by setting $[\alpha]_0 \in_R \mathbb{Z}_q$ and $[x]_\kappa = [\delta]_i$. Then $[\alpha * x]_h = \mathcal{E}([x]_\kappa, [\alpha]_{h-\kappa})$. Given the output of a successful adversary $\mathcal{A}$ to EDP, $\mathcal{A}([\alpha]_0, [\alpha * x]_h, [x]_\kappa) \rightarrow [x]_j$ for $j < \kappa$, we return $[x]_j$ as the

answer to the EDP instance.
Therefore the probability of success against DP is identical to $\mathcal{A}$ against EDP. □

For the efficiency and correctness of our scheme we let $h = (l)$, where l is the size of the map, and $\kappa \in (1, ..., l)$.

5.3.2 Major Issues

Three major multilinear map candidates are [GGH13][CLT13][GGH15]. They have been the targets of many attacks and patches.

Several attacks on multilinear maps are called "zeroising" attacks; they run in polynomial-time but require the availability of low-level encodings of zero [GGH13][HJ16]. There are also sub-exponential and quantum attacks [CDPR16][ABD16][CJL16]. Further to this, recently Miles *et al* introduced a class of "annihilation" attacks on multilinear maps [MSZ16].

There are reasons to believe that multilinear maps may be unrealisable. In particular, the near equivalence to indistinguishable obfuscation [PS15]—an extremely powerful tool which (in its stronger variant) is known not to exist [BGI$^+$12]—is worrying. Furthermore, Boneh and Silverberg [BS03] in their original paper on the applications of multilinear maps present several results which cast doubt on the multilinear maps as an "unlikely-ability" and they concluded that "such maps might have to either come from outside the realm of algebraic geometry, or occur as unnatural computable maps arising from geometry."

We will leave aside here all questions of the computational efficiency of multilinear map candidates, though this would be a significant concern for any implementation.

5.4 Construction

In this section we discuss our construction. First we provide the intuition for the construction, then go through the details of each component algorithm. We use the multilinear map notation presented in subsection 5.3.

5.4.1 Intuition

Before discussing the construction in detail, we briefly describe the intuition of the construction.

To gain unconditional anonymity (information theoretic privacy) we make use of the Pederson commitment which provides unconditional hiding of the private key in the public key. We then make use of a multilinear map to raise the key at each time period, which provides forward security.

In the signature we use the Fiat-Shamir heuristic on two knowledge of discrete logs proofs, combined into one. The signer proves firstly that they know x behind $f = dx$, and secondly that they know x and y such that $gx + hy$ is one of the public keys, where the x in the first and second parts are the same. The proofs in verification and signing also include all other public keys and random challenges c_i. Since both the real challenge c and random challenges c_i are distributed uniformly at random, an adversary is unable to discern which party signed.

5.4.2 Setup(n)

Take as input the number of time periods $\mathcal{T}$. We will let $t \in (0, ..., \mathcal{T} - 1)$ denote the current time period. Run the setup algorithm for a multilinear map, to construct a $(\mathcal{T} + 1)$-linear map denoting its public parameters **params**, we will refer to the maximum allowed level as l. Let H_i denote the ith element in a family of hash functions H such that H_i: $\{0,1\}^* \rightarrow [\mathbb{G}]_i$. Construct $[g]_0 = H_0$("Generator-g") and $[h]_l = H_l$("Generator-h"). The public parameters are **param** = (**params**, $[g]_0, [h]_l, H$,"Generator-g","Generator-h"). Note that everyone can check the generation of g, h.

5.4.3 Key Generation

A user randomly chooses $[x]_0, [y]_0 \in_R [\mathbb{G}]_0$ and computes $[Z]_l = \mathcal{E}(\mathcal{E}([g]_0, [x]_0), [1]_l) + \mathcal{E}([h]_l, [y]_0)$. His secret key is $sk = ([x]_0, [y]_0)$ and the corresponding public key is $pk = [Z]_l$.

5.4.4 Sign

On input $(event, n, \vec{pk}_t, sk_\pi, M, t)$, where $event$ is the event description, n is the number of users included in the ring signature, $\vec{pk}_t = \{pk_i, ..., pk_n\} = \{[Z_1]_l, ..., [Z_n]_l\}$ is the set of public

keys of users in the ring, sk_π is the secret key corresponding to the public key pk_π such that $pk_\pi \in \vec{pk}_t$ (w.l.o.g., $\pi \in [1, n]$), M is the message to be signed, and t is the time period, the user (with the knowledge of $sk_\pi = ([x]_t, [y]_0)$) computes the following:

1. Compute $[d]_{l-t-1} = H_{l-t-1}(t||event)$ and $[f]_{l-1} = \mathcal{E}([d]_{l-t-1}, [x]_t)$
2. Randomly generate $[r_x]_t \in_R [\mathbb{G}]_t$ and $[c_1]_0, ..., [c_{\pi-1}]_0, [c_{\pi+1}]_0, ..., [c_n]_0, [r_y]_0 \in_R [\mathbb{G}]_0$ and compute

$$[K]_l = \mathcal{E}([g]_{l-t}, [r_x]_t) + \mathcal{E}([h]_l, [r_y]_0) + \sum_{i=1, i\neq\pi}^{n} \mathcal{E}([Z_i]_l, [c_i]_0),$$

$$[K']_{l-1} = \mathcal{E}([d]_{l-t-1}, [r_x]_t) + \mathcal{E}([f]_{l-1}, \sum_{i=1, i\neq\pi}^{n} [c_i]_0)$$

3. Find $[c_\pi]_0$ such that

$$[c_\pi]_0 : H_0(\vec{pk}_t||event||[f]_{l-i}||M||[K]_l||[K']_{l-1}||t) - \sum_{i=1, i\neq\pi}^{n} [c_i]_0$$

4. Compute

$$[\tilde{x}]_t = [r_x]_t - \mathcal{E}([c_\pi]_0, [x]_t), \qquad [\tilde{y}]_0 = [r_y]_0 - \mathcal{E}([c_\pi]_0, [y]_0)$$

5. Output the signature $\sigma = ([f]_{l-1}, [\tilde{x}]_t, [\tilde{y}]_0, [c_1]_0, ..., [c_n]_0)$.

5.4.5 Verify

On input $(event, n, \vec{pk}_t, M, \sigma, t)$, first compute $[d]_{l-t-1} = H_{l-t-1}(t||event)$ and

$$[c_0]_0 = H_0((\vec{pk}_t||event||[f]_{l-1}||M||\mathcal{E}([g]_{l-t}, [\tilde{x}]_t) + \mathcal{E}([h]_l, [\tilde{y}]_0) + \sum_{i=1}^{n} \mathcal{E}([Z_i]_l, [c_i]_0)||\mathcal{E}([d]_{l-t-1}, [\tilde{x}]_t) + \mathcal{E}([f]_{l-1}, \sum_{i=1}^{n} [c_i]_0)||t))$$

then check whether

$$\sum_{i=1}^{n} [c_i]_0 = [c_0]_0$$

5.4.6 Link

On input two signatures $\sigma_1 = ([f_1]_{l-1}, *), \sigma_2 = ([f_2]_{l-1}, *)$, two messages M_1, M_2, an event description $event$, and a time t, first check whether the two signatures are valid. If yes, output **linked** if $[f_1]_{l-1} = [f_2]_{l-1}$ and output **unlinked** otherwise.

5.4.7 Private Key Update

In a given time period t to calculate the private key for time $t+1$, do:

$$[x]_{t+1} = \mathcal{E}([1]_1, [x]_t)$$

5.4.8 Public Key Update

In a given time period t to calculate the public key for time period $t+1$, do:[5]

$$[Z]_l = [Z]_l$$

5.5 Correctness

In this section we show that our scheme satisfies our various requirements for correctness. We focus on security in a later section.

5.5.1 Verification Correctness

To show that the above scheme satisfies verification correctness, it suffices to show that the verification values K and K' calculated by each party are the same.

Proof. We first show that the K calculated by the signer, K_s, is the same as that calculated by

[5]This is a null operation which does nothing. We write it out only for the purpose of making a clear comparison with other schemes.

the verifier, K_v:

$$
\begin{aligned}
[K_s]_l &= [g * r_x]_l + [h * r_y]_l + \sum_{i=1, i\neq\pi}^{n} [Z_i * c_i]_l \\
[K_v]_l &= [g * \tilde{x}]_l + [h * \tilde{y}]_l + \sum_{i=1}^{n} [Z_i * c_i]_l \\
&= [g * \tilde{x}]_l + [h * \tilde{y}]_l + [Z_\pi * c_\pi]_l + \sum_{i=1, i\neq\pi}^{n} [Z_i * c_i]_l \\
&= [g(r_x - c_\pi x)]_l + [h(r_y - c_\pi y)]_l + [g(c_\pi x) + h(c_\pi y)]_l + \sum_{i=1, i\neq\pi}^{n} [Z_i * c_i]_l \\
&= [g * r_x]_l + [h * r_y]_l + \sum_{i=1, i\neq\pi}^{n} [Z_i * c_i]_l \qquad \square
\end{aligned}
$$

Proof. Now we show that K'_s is equal to K'_v:

$$
\begin{aligned}
[K'_s]_{l-1} &= [d * r_x]_{l-1} + [f * \sum_{i=1, i\neq\pi}^{n} c_i]_{l-1} \\
[K'_v]_{l-1} &= [d * \tilde{x}]_{l-1} + [f * \sum_{i=1}^{n} c_i]_{l-1} \\
&= [d * \tilde{x}]_{l-1} + [f * c_\pi]_{l-1} + [f * \sum_{i=1, i\neq\pi}^{n} c_i]_{l-1} \\
&= [d(r_x - c_\pi x)]_{l-1} + [d(c_\pi x)]_{l-1} + [f * \sum_{i=1, i\neq\pi}^{n} c_i]_{l-1} \\
&= [d * r_x]_{l-1} + [f * \sum_{i=1, i\neq\pi}^{n} c_i]_{l-1} \qquad \square
\end{aligned}
$$

5.5.2 Linking Correctness

For a given event $event$, time t, and private key $[x]_t$ the linking component, $[d]_{l-t-1} = H_{l-t-1}(t||event)$, $[f]_{l-1} = \mathcal{E}([d]_{l-t-1}, [x]_t)$, is completely deterministic. Since the linking component is completely deterministic, under the above conditions, given any two signatures a simple equality check on the linking component suffices.

Conversely, for a given event $event$, time t, and two different private keys $[x]_t$ and $[x']_t$ the linking element will be different[6].

[6]While it is possible for two different private keys to have the same public key, which would break the asseration above, this breaks the Pedersen commitments and reveals the relationship between g and h. It is also possible with

5.5.3 Update Correctness

For a given time period t to calculate the updated keys for the next time period using the update function, we observe that the relation between public and private keys is unchanged. Recall that the use of the pairing produces the product of the input values at the level of the sum of the input levels. The only changes to the private keys and public key is $[x]_{t+1} = \mathcal{E}([1]_1, [x]_t)$, which simply "raises" up one floor without changing the encoded value.

5.6 Security Analysis

The security proof is in the ROM using the forking lemma [PS96b].

Theorem 18. *Our forward secure linkable ring signature is unforgeable in the random oracle model, if EDP is hard.*

Liu et al reduced the security of unforgeability to the discrete log by rewinding for all $[c]_i$, in the worst case. Therefore, the success of their simulation diminishes exponentially in n, the number of users in the ring. We present a new proof which extracts using $\sum_{i=1}^{n}[c_i]_0$ which means we only have to rewind once. Our reduction is hence independent of n rather than exponential. We also embed the challenge in all public keys rather just than a subset.

Proof. Given an (l, l)-EDP instance $([\alpha]_0, [\alpha * x]_l, [x]_l)$, $\mathcal{B}$ is asked to output $[x]_j$ where $j < l$, note that the secret key x for all time periods in our scheme satisfies this bound. $\mathcal{B}$ sets $[h]_l = [\alpha * x]_l$ and $[g]_0 = [\alpha]_0$.

Oracle Simulation: $\mathcal{B}$ simulates the oracles as follows.

- *Random Oracles* H_i: For query input H_0("GENERATOR-g"), $\mathcal{B}$ returns $[g]_0$. For query input H_l("GENERATOR-h"), $\mathcal{B}$ returns $[h]_l$. For other queries, $\mathcal{B}$ randomly picks $[\lambda]_0 \in_R [\mathbb{G}]_0$, sets $[a]_i = \mathcal{E}([\lambda]_0, [1]_i)$ and returns $[a]_i$.

- *Joining Oracle* $\mathcal{JO}$: $\mathcal{B}$ chooses $[x]_0, [y]_0 \in_R [\mathbb{G}]_0$ and sets $[Z]_l = \mathcal{E}([gx]_0, [1]_l) + \mathcal{E}([h]_l, [y]_0)$. $\mathcal{B}$ stores the tuple $([Z]_l, [x]_0, [y]_0)$.

negligible probability for the hash function to collide.

- *Corruption Oracle $\mathcal{CO}$*: On input a public key pk which is an output from $\mathcal{JO}$, $\mathcal{B}$ outputs the corresponding private key.

- *Signing Oracle $\mathcal{SO}$*: On input a signing query for event $event$, a set of public keys $\vec{pk}_t = \{[Z_1]_l, ..., [Z_n]_l\}$, the public key for the signer $[Z_\pi]_l$, where $\pi \in [1, n]$, and a message M, time t, $\mathcal{B}$ simulates as follows:

  - If the query of $H_{l-t-1}(t||event)$ has not been made, carry out the H-query of $t||event$ as described above. Set $[d]_{l-t-1}$ to $H_{l-t-1}(t||event)$. Note that $\mathcal{B}$ knows the $[\lambda]_0$ that corresponds to $[d]_{l-t-1}$.

  - Since $\mathcal{B}$ knows the private keys for all possible π it simply constructs σ according to the scheme.

  - $\mathcal{B}$ returns the signature $\sigma = ([f]_{l-1}, [\tilde{x}]_t, [\tilde{y}]_0, [c_1]_0, ..., [c_n]_0)$. $\mathcal{A}$ cannot distinguish between $\mathcal{B}$'s simulation and real life.

Output: For one successful simulation, suppose the forgery of $\mathcal{A}$ is $\sigma^1 = ([f^1]_{l-1}, [\tilde{x}^1]_t, [\tilde{y}^1]_0, [c_1^1]_0, ..., [c_{n'}^1]_0)$ on an event $event$, time t and a set of public keys $\vec{pk}_t''$. By the assumption of random oracle model, $\mathcal{A}$ has a query $H_{l-t-1}(t||event)$ which is denoted by $[d]_{l-t-1}$ and query $H_0(\vec{pk}_t||event||[f]_{l-1}||M||[K]_l||[K']_{l-1}||t)$ where:

$$[K]_l = \mathcal{E}([g]_{l-t}, [\tilde{x}^1]_t) + \mathcal{E}([h]_l, [\tilde{y}^1]_0) + \textstyle\sum_{i=1}^{n} \mathcal{E}([Z_i]_l, [c_i^1]_0) \text{ and}$$
$$[K']_{l-1} = \mathcal{E}([d]_{l-t-1}, [\tilde{x}^1]_t) + \mathcal{E}([f^1]_{l-1}, \textstyle\sum_{i=1}^{n} [c_i^1]_0)$$

After a successful rewind we get another signature $\sigma^2 = ([f^1]_{l-1}, [\tilde{x}^2]_t, [\tilde{y}^2]_0, [c_1^2]_0, ..., [c_{n'}^2]_0)$ note that $[f^1]_{l-1}, [K]_l, [K']_l$ and $\vec{pk}_t$ must be the same, since we rewind only to the point of the H_0 query. Note also that in the rewind we force a change in the H_0 oracle output to the query $\sum_{i=1}^{n}[c_i]$.

Recall that for $i = [1, 2]$, the following equations hold by the definitions of $\tilde{x}$, H_0 and K':

$$\begin{aligned}
[\tilde{x}^i]_t &= [r_x^i]_t - [c_\pi^i x]_t \\
[\tilde{x}^i]_{l-1} &= \frac{[K']_{l-1}}{[d]_0} - [x \sum_{k=1;i\neq\pi}^{n} c_k^i]_{l-1} - [c_\pi^i x]_{l-1} \\
&= \frac{[K']_{l-1}}{[d]_0} - [x \sum_{k=1}^{n} c_k^i]_{l-1} \\
&= \frac{[K']_{l-1}}{[d]_0} - [x H_0^i]_{l-1}
\end{aligned}$$

We can now see that $[\tilde{x}]$, for a given x, must change when $[H_0^i]_{l-1}$ changes. Therefore we have $[\tilde{x}^1] \neq [\tilde{x}^2]$. We can then calculate the challenge $[\gamma]_{l-1}$ as;

$$\begin{aligned}
[g\gamma]_{l-1} &= \frac{([g\tilde{x}^1]_{l-1} + \sum_{i=1}^{n}[gx_i]_{l-1}[c_i^1]_{l-1}) - ([g\tilde{x}^2]_{l-1} + \sum_{i=1}^{n}[gx_i]_{l-1}[c_i^2]_{l-1}))}{[\tilde{y}^2]_0 - [\tilde{y}^1]_0 - \sum_{i=1}^{n}[y_i]_0[c_i^1 - c_i^2]_0} \\
[g\gamma]_{l-1} &= \frac{([K]_{l-1} - [\tilde{y}^1 h]_{l-1} - \sum_{i=1}^{n}[hy_i]_{l-1}[c_i^1]_{l-1}) - ([K]_{l-1} - [\tilde{y}^2 h]_{l-1} - \sum_{i=1}^{n}[hy_i]_{l-1}[c_i^2]_{l-1})}{[\tilde{y}^2]_0 - [\tilde{y}^1]_0 - \sum_{i=1}^{n}[y_i]_0[c_i^1 - c_i^2]_0} \\
[g\gamma]_{l-1} &= \frac{-[\tilde{y}^1 h]_{l-1} - \sum_{i=1}^{n}[hy_i]_{l-1}[c_i^1]_{l-1} + [\tilde{y}^2 h]_{l-1} + \sum_{i=1}^{n}[hy_i]_{l-1}[c_i^2]_{l-1}}{[\tilde{y}^2]_0 - [\tilde{y}^1]_0 - \sum_{i=1}^{n}[y_i]_0[c_i^1 - c_i^2]_0} \\
[g\gamma]_{l-1} &= \frac{-[\tilde{y}^1 h]_{l-1} - \sum_{i=1}^{n}[hy_i]_{l-1}[c_i^1 - c_i^2]_{l-1} + [\tilde{y}^2 h]_{l-1}}{[\tilde{y}^2]_0 - [\tilde{y}^1]_0 - \sum_{i=1}^{n}[y_i]_0[c_i^1 - c_i^2]_0} \\
[g\gamma]_{l-1} &= \frac{[h]_{l-1}(-[\tilde{y}^1]_{l-1} - \sum_{i=1}^{n}[y_i]_{l-1}[c_i^1 - c_i^2]_{l-1} + [\tilde{y}^2]_{l-1})}{[\tilde{y}^2]_0 - [\tilde{y}^1]_0 - \sum_{i=1}^{n}[y_i]_0[c_i^1 - c_i^2]_0} \\
[g\gamma]_{l-1} &= [h]_{l-1}
\end{aligned}$$

Recall that we set $[g]_0 = [\alpha]_0$ and $[h]_l = [\alpha * x]_l$, we have now learned γ such that $[g\gamma]_{l-1} = [h]_{l-1}$, in other words $[\gamma]_{l-1}$ is equal to the challenge $[x]_{l-1}$ which we now successfully return.

By the forking lemma, the chance of each successful rewind simulation is at least $\xi/4$, where ξ is the probability that $\mathcal{A}$ successfully forges a signature. Hence the probability that for a given adversary $\mathcal{A}$, we can extract $[x]_{l-1}$ is at least $\frac{\xi}{4}$.

□

Then, we prove our scheme is unconditionally anonymous.

Theorem 19. *Our forward secure linkable ring signature is unconditionally anonymous.*

Proof. Note that the proof of unconditional anonymity is largely unchanged from [LASZ14], since both schemes rely on Pederson commitments. For each $\mathcal{JO}$ query, a value $[Z]_l = \mathcal{E}([g]_0, [x]_0) + \mathcal{E}([h]_l, [y]_0)$ is returned for some random pair $([x]_0, [y]_0)$. The challenge signature is constructed using the key of a random signer in the ring. In what follows, we are going to show that the advantage of the adversary is information-theoretically zero. The proof is divided into three parts. First, we show that given a signature $\sigma = ([f]_{l-1}, [\tilde{x}]_t, [\tilde{y}]_0, [c_1]_0, ..., [c_\pi]_0)$ for a ring of public keys $([Z_1]_l, ..., [Z_n]_l)$ on message M, event $event$ and time t, there exists a corresponding private key $([x_\pi]_t, [y_\pi]_0)$ for each possible public key $[Z_\pi]_l$, for any $\pi \in \{1, ..., n\}$, that can construct the linking tag $[f]_{l-1}$. That is, $[f]_{l-1} = \mathcal{E}(H_{l-t-1}(t||event), [x_\pi]_t)$. In what follows, we use $[d]_{l-t-1}$ to denote the value of $H_{l-t-1}(t||event)$. Second, we show that given such a private key $([x_\pi]_t, [y_\pi]_0)$ there exists a tuple $([r_{x_\pi}]_t, [r_{y_\pi}]_0)$ so that σ corresponds to $([x_\pi]_t, [y_\pi]_0)$ using randomness $([r_{x_\pi}]_t, [r_{y_\pi}]_0)$. Finally, we show that for any value of $\pi \in \{1, ..., n\}$, the distribution of the tuple $([x_\pi]_t, [y_\pi]_0, [r_{x_\pi}]_t, [r_{y_\pi}]_0)$ defined in parts one and two is identical. Therefore—in the view of the adversary—the signature σ is independent to the value π, the index of the actual signer. Thus, we can conclude that even an unbounded adversary cannot output the value π with probability better than guessing:

1. *Part I.* Let x, y be the values so that $[f]_{l-1} = \mathcal{E}([d]_{l-t-1}, [x]_t)$ and $[g]_0 = \mathcal{E}([h]_0, [y]_0)$. Further, let $[Z_i]_l = \mathcal{E}([h]_0, [z_i]_l)$ for $i = 1$ to n. For each $\pi \in \{1, ..., n\}$, consider the values

$$[x_\pi]_t = [x]_t,$$

$$[y_\pi]_t = [z_\pi]_t - \mathcal{E}([x_\pi]_t, [y]_0)$$

   Obviously, $([x_\pi]_t, [y_\pi]_t)$ is a private key corresponding to the public key $[Z_\pi]_l$ (since $[Z_\pi]_l = \mathcal{E}([h]_{l-t}, [z_\pi]_t) = \mathcal{E}([h]_{l-t}, \mathcal{E}([x_\pi]_t, [y]_0) + [y_\pi]_t) = \mathcal{E}([g]_{l-t}, [x_\pi]_t) + \mathcal{E}([h]_{l-t}, [y_\pi]_t))$ and $[f]_{l-1} = \mathcal{E}([d]_{l-t-1}, [x]_t]) = \mathcal{E}([d]_{l-t-1}, [x_\pi]_t)$.

2. *Part II.* For each possible $([x_\pi]_t, [y_\pi]_t)$ defined in Part I, consider the values

$$[r_{x_\pi}]_t := [\tilde{x}]_t + \mathcal{E}([c_\pi]_0, [x_\pi]_t),$$

$$[r_{y_\pi}]_t := [\tilde{y}]_t + \mathcal{E}([c_\pi]_0, [y_\pi]_t),$$

It can be seen that σ can be created by the private key $([x_\pi]_t, [y_\pi]_t)$ using the randomness $([r_{x_\pi}]_t, [y_{y_\pi}]_t)$, for any $\pi \in \{1, ..., n\}$.

3. *Part III.* It is straightforward to show that the distribution of $([x_\pi]_t, [y_\pi]_t, [r_{x_\pi}]_t, [y_{y_\pi}]_t)$ for each possible value π are identical and that it adheres to the distribution to a signature created by the signer with public key $[Z_\pi]_l$.

In other words, the signatures σ can be created by any signer equipped with private key $([x_\pi]_t, [y_\pi]_t)$ for any $\pi \in \{1, ..., n\}$ using randomness $([r_{x_\pi}]_t, [y_{y_\pi}]_t)$. Even if the unbounded adversary can compute all the values $([x_\pi]_t, [y_\pi]_t, [r_{x_\pi}]_t, [y_{y_\pi}]_t)$ for all $\pi = 1$ to n, it still cannot guess who the actual signer is.

We are using the fact that a public key in our construction corresponds to multiple secret keys. For each public key, there exists a unique corresponding private key that would match with the given linking tag in the signature. □

Now, we prove our scheme is linkable.

Theorem 20. *Our forward secure linkable ring signature is linkable in the random oracle model, if the EDP is hard.*

Proof. If $\mathcal{A}$ can produce two valid signatures that are unlinked with just one private key, we can use this to successfully break EDP. We use the same setting as the proof in Theorem 18, with the exception that the adversary is given a pair (x, y) valid for $[Z] \in \vec{pk}_t$ as an output of the corruption oracle.

If given a pair of $\sigma^i = ([f^i]_{l-1}, [\tilde{x}^i]_t, [\tilde{y}^i]_0, [c_1^i]_0, ..., [c_{n'}^i]_0)$ on an event $event$, time t and a set of public keys $\vec{pk}_t''$, then by the assumption of random oracle model, $\mathcal{A}$ has made two queries $H_{l-t-1}^i(t||event)$ which is denoted by $[d^i]_{l-t-1}$ and two queries $H_0^i(\vec{pk}_t||event||[f^i]_{l-1}||M||[K^i]_l||[K'^i]_{l-1}||t)$ where

$$[K]_l = \mathcal{E}([g]_{l-t}, [\tilde{x}^1]_t) + \mathcal{E}([h]_l, [\tilde{y}^1]_0) + \textstyle\sum_{i=1}^n \mathcal{E}([Z_i]_l, [c_i^1]_0) \text{ and}$$
$$[K']_{l-1} = \mathcal{E}([d]_{l-t-1}, [\tilde{x}^1]_t) + \mathcal{E}([f^1]_{l-1}, \textstyle\sum_{i=1}^n [c_i^1]_0)$$

Since $\sigma^1 \neq \sigma^2$ and they are unlinked, by definition of linkability we have $[f^1]_{l-1} \neq [f^2]_{l-1}$. Since, by definition of the game, the σ^i are both valid for the same time and event, $[d^1]_{l-t-1} =$

$H_{l-t-1}(t||event) = [d^2]_{l-t-1}$. Recall that $[f^i]_{l-1} = [d^1 x^i]_{l-1}$, where we have shown $[d^1]_{l-t-1} = [d^2]_{l-t-1}$. Hence, $[x^1]_t \neq [x^2]_t$. Therefore at most one $[f]_{l-1}$, and hence σ^i, encodes the pair (x, y) which we gave to the adversary. We use the method from Theorem 18 to extract $[\gamma]_{l-1}$ from the other signature σ'.

The probability that, for a given adversary $\mathcal{A}$, we can extract $[x]_{l-1}$ is at least $\frac{\xi}{4}$.

□

Now, we prove our scheme is non-slanderable.

Theorem 21. *Our forward secure linkable ring signature is non-slanderable in the random oracle model, if EDP is hard.*

Proof. We use the same setting as Theorem 18. $\mathcal{A}$ can query any oracle except that it cannot submit a chosen public key pk_π to the $\mathcal{CO}$. It then gives $\mathcal{B}$ pk_π, a list of public keys $\vec{pk}_t$ (w.l.o.g, we have $|\vec{pk}_t| = n$) such that $pk_\pi \in \vec{pk}_t$, a message M, an event description $event$, and a time t. In return, simulator $\mathcal{B}$ generates a signature $\sigma([f]_{l-1}, .)$ using the standard method for the joining oracle, and gives it back to $\mathcal{A}$. Since we choose $[f]_{l-1} = [dx]_{l-1}$ at random for a fixed d we have implicitly defined $[x]_t$ at random. $\mathcal{A}$ continues to query various oracles, expect that it is not allowed to submit pk_π to $\mathcal{CO}$.

Suppose $\mathcal{A}$ produces another valid signature $\sigma^* = ([f']_{l-1}, .)$ such that it is not an output for $\mathcal{SO}$ and it is linkable to σ. Since they are linkable, we have $[f']_{l-1} = [f]_{l-1}$ and hence $\frac{[x]_{l-1}}{[d]_0} = \frac{[x]_{l-1}}{[d]_0}$. Recall that, by definition of the game, $\sigma^* \neq \sigma'$ which implies that $[\tilde{x}^*] \neq [\tilde{x}]'$ and hence $[\tilde{y}^*] \neq [\tilde{y}']$.

Then extract $[\gamma]_{l-1}$ from σ^* as outlined in Theorem 18. The probability that, for a given adversary $\mathcal{A}$, we can extract $[x]_{l-1}$ is $\frac{\xi}{4}$. □

Finally, we prove our scheme is forward secure.

Theorem 22. *Our forward secure linkable ring signature is forward secure in the random oracle model, if EDP is hard.*

Proof. We show that the ability of the adversary to make corruption queries at times later than t does not allow it to calculate the private key at time t without breaking $(l, \kappa = t + 1)$-EDP and

hence the system achieves forward security, for $\kappa \in (1, l-1)$. In this proof we start by guessing the break point t for which the adversary's σ will be valid.

Given an (κ, l)-EDP instance $([\alpha]_0, [\alpha * x]_l, [x]_\kappa)$, $\mathcal{B}$ is asked to output $[x]_j$ where $j < \kappa$. $\mathcal{B}$ picks $[h]_0 \in_R [\mathbb{G}]_0$ and sets $[h]_l = \mathcal{E}([h]_0, [1]_l)$. $\mathcal{B}$ also chooses $[y]_0 \in_R [\mathbb{G}]_0$ and sets $[Z]_l = [\alpha * x]_l + (\mathcal{E}([h]_l, [y]_0)$.

Oracle Simulation: $\mathcal{B}$ simulates the oracles as follows.

- *Random Oracles* H_i: For query input H_0("GENERATOR-g"), $\mathcal{B}$ returns $[\alpha]_0$. For query input H_l("GENERATOR-h"), $\mathcal{B}$ returns $[h]_l$. For other queries, $\mathcal{B}$ randomly picks $[\lambda]_0 \in_R [\mathbb{G}]_0$, sets $[a]_i = \mathcal{E}([\lambda]_0, [1]_i)$ and returns $[a]_i$.

- *Joining Oracle* $\mathcal{JO}$: Assume $\mathcal{A}$ can only query $\mathcal{JO}$ for a maximum n' times, where $n' = n+1$. W.l.o.g., we use $(1, ..., n)$ to denote the indexes for which $\mathcal{B}$ knows the private keys and n' to denote the challenge index. For the first n indexes, $\mathcal{B}$ generates the public key and private key pair according to the algorithm. Upon the jth query, $\mathcal{B}$ returns the corresponding public key.

- *Corruption Oracle* On input a public key pk_i which is an output from $\mathcal{JO}$, and a time t, $\mathcal{B}$ checks whether it is corresponding to $[1, n]$, if yes, then $\mathcal{B}$ returns the private key. Otherwise, if time $t \geq \kappa$, $\mathcal{B}$ returns $sk_i = ([x_i]_t, [y_i]_t)$ at time t, otherwise $\mathcal{B}$ halts.

- *Signing Oracle* $\mathcal{SO}$: On input a signing query for event $event$, a set of public key $\vec{pk}_t = \{[Z_1]_l, ..., [Z_n]_l\}$, the public key for the signer $[Z_\pi]_l$, where $\pi \in [1, n]$, and a message M, and time t, $\mathcal{B}$ simulates as follows:

  - If the query of $H_{l-t-1}(t||event)$ has not been made, carry out the H-query of $t||event$ as described above. Set $[d]_{l-t-1}$ to $H_{l-t-1}(t||event)$. Note that $\mathcal{B}$ knows the $[\lambda]_0$ that corresponds to $[d]_{l-t-1}$.
  - If $[Z_\pi]_l$ is not corresponding to n', $\mathcal{B}$ knows the private key and computes the signature according to the algorithm. Otherwise, B sets $[f]_{l-1} = [dx]_{l-1}$.
  - $\mathcal{B}$ randomly chooses $[\tilde{x}]_t \in_R [\mathbb{G}]_t$ and $[c_i]_0, [\tilde{y}]_0 \in_R [\mathbb{G}]_0$ for all $i \in [1, n]$ and sets the H_0 oracle output of

$$H_0(\vec{pk}_t||event||[f]_{l-1}||M||\mathcal{E}([g]_{l-t}, [\tilde{x}]_t) + \mathcal{E}([h]_{l-t}, [\tilde{y}]_t) + \sum_{i=1}^{n} \mathcal{E}([Z_i]_l, [c_i]_0)||\mathcal{E}([d]_{l-t-1}, [\tilde{x}]_t) + \mathcal{E}([f]_{l-1}, \sum_{i=1}^{n} c_i)||t)$$

- $\mathcal{B}$ returns the signature $\sigma = ([f]_{l-1}, [\tilde{x}]_t, [\tilde{y}]_t, [c_1]_0, ..., [c_n]_0)$. $\mathcal{A}$ cannot distinguish between $\mathcal{B}$'s simulation and real life.

Output: For one successful simulation, suppose the forgery of $\mathcal{A}$ is $\sigma^1 = ([f^1]_{l-1}, [\tilde{x}^1]_t, [\tilde{y}^1]_0, [c_1^1]_0, ..., [c_{n'}^1]_0)$ on an event $event$, time t and a set of public keys $\vec{pk}_t''$. By the assumption of random oracle model, $\mathcal{A}$ has a query $H_{l-t-1}(t||event)$ which is denoted by $[d]_{l-t-1}$ and query $H_0(\vec{pk}''||event||[f]_{l-1}||M||[K]_l||[K']_{l-1}||t)$ where

$$[K]_l = \mathcal{E}([g]_{l-t}, [\tilde{x}^1]_t) + \mathcal{E}([h]_l, [\tilde{y}^1]_0) + \textstyle\sum_{i=1}^n \mathcal{E}([Z_i]_l, [c_i^1]_0) \text{ and}$$
$$[K']_{l-1} = \mathcal{E}([d]_{l-t-1}, [\tilde{x}^1]_t) + \mathcal{E}([f^1]_{l-1}, \textstyle\sum_{i=1}^n [c_i^1]_0)$$

After a successful rewind we get another signature $\sigma^2 = ([f^1]_{l-1}, [\tilde{x}^2]_t, [\tilde{y}^2]_0, [c_1^2]_0, ..., [c_{n'}^2]_0)$. Note that $[f^1]_{l-1}$ and the $[K]_l, [K']_l$ must be the same, since we rewind only to the point of lth query. Note also that in the rewind we forced a change in $H_0 = \sum_{i=1}^n [c_i]$
Recall that for $i = [1, 2]$, the following equations hold by the definitions of $\tilde{x}, H$ and K':

$$\begin{aligned}
[\tilde{x}^i]_t &= [r_x^i]_t - [c_\pi^i x']_t \\
[\tilde{x}^i]_{l-1} &= \frac{[K']_{l-1}}{[d]_0} - [x' \sum_{k=1; i\neq\pi}^{n} c_k^i]_{l-1} - [c_\pi^i x']_{l-1} \\
&= \frac{[K']_{l-1}}{[d]_0} - [x' \sum_{k=1}^{n} c_k^i]_{l-1} \\
&= \frac{[K']_{l-1}}{[d]_0} - [x' H^i]_{l-1}
\end{aligned}$$

We can now see that we have two commitments to $[x']_{l-1}$ for a fixed $\frac{[K']_{l-1}}{[d]_0}$ which we know and for different H^i which we know. We can therefore calculate $[x']_t$ as follows:

$$\begin{aligned}
\frac{[\tilde{x}^1]_t - [\tilde{x}^2]_t}{-[H^1]_0 + [H^2]_0} &= \frac{\frac{[K']_{l-1}}{[d]_0} - [x'H^1]_{l-1} - \frac{[K']_{l-1}}{[d]_0} + [x'H^2]_{l-1}}{-[H^1]_0 + [H^2]_0} \\
&= \frac{[x']_t(-[H^1]_0 + [H^2]_0)}{-[H^1]_0 + [H^2]_0} \\
&= [x']_t
\end{aligned}$$

Recall that:

$$
\begin{aligned}
[\tilde{y}^i]_0 &= [r_y^i]_0 - [c_\pi^i y]_0 \\
[\tilde{y}^i]_0 &= \frac{[K]_t - [gr_x]_t - \sum_{k=1;k\neq\pi}^{n}[Z_k c_k^i]}{[h]_0} - [c_\pi^i \phi]_t \\
[\tilde{y}^i]_0 + \frac{[\tilde{x}]}{[h]_0} &= \frac{[K]_t - [gc_\pi x]_t - \sum_{k=1;k\neq\pi}^{n}[Z_k c_k^i]}{[h]_0} - [c_\pi^i \phi]_t \\
[\tilde{y}^i]_0 + \frac{[\tilde{x}]}{[h]_0} &= \frac{[K]_t - [gc_\pi x]_t - \sum_{k=1;k\neq\pi}^{n}[gx_k c_k^i]_t - \sum_{k=1;k\neq\pi}^{n}[hy_k c_k^i]_t}{[h]_0} - [c_\pi^i \phi]_t \\
[\tilde{y}^i]_0 + \frac{[\tilde{x}]}{[h]_0} &= \frac{[K]_t - \sum_{k=1}^{n}[gx_k c_k^i]_t - \sum_{k=1;k\neq\pi}^{n}[hy_k c_k^i]_t}{[h]_0} - [c_\pi^i \phi]_t \\
[\tilde{y}^i]_0 + \frac{[\tilde{x}]}{[h]_0} + \frac{\sum_{k=1}^{n}[gx_k c_k^i]_t}{[h]_0} &= \frac{[K]_t - \sum_{k=1;k\neq\pi}^{n}[hy_k c_k^i]_t}{[h]_0} - [c_\pi^i \phi]_t \\
[\tilde{y}^i]_0 + \frac{[\tilde{x}]}{[h]_0} + \frac{\sum_{k=1}^{n}[gx_k c_k^i]_t}{[h]_0} &= \frac{[K]_t - \sum_{k=1}^{n}[hy_k c_k^i]_t}{[h]_0} \\
[\tilde{y}^i]_0 + \frac{[\tilde{x}]}{[h]_0} + \frac{\sum_{k=1}^{n}[gx_k c_k^i]_t}{[h]_0} &= \frac{[K]_t}{[h]_0} - \sum_{k=1}^{n}[y_k c_k^i]_t \\
[\tilde{y}^i]_0 + \frac{[\tilde{x}]}{[h]_0} + \frac{\sum_{k=1}^{n}[gx_k c_k^i]_t}{[h]_0} &= \frac{[K]_t}{[h]_0} - \sum_{k=1}^{n}[y_k]_t \sum_{k=1}^{n}[c_k^i]_t \\
[\tilde{y}^i]_0 + \frac{[\tilde{x}]}{[h]_0} + \frac{\sum_{k=1}^{n}[gx_k c_k^i]_t}{[h]_0} &= \frac{[K]_t}{[h]_0} - \sum_{k=1}^{n}[y_k]_t [H^i]_t
\end{aligned}
$$

Note that the final form on a left hand side is calculable; on the right, a constant minus a constant multiplied by H. We can hence find $\sum_{\kappa=1}^{n}[y_\kappa]_t$ as:

$$
\sum_{\kappa=1}^{n}[y_\kappa]_t = \frac{([\tilde{y}^1]_0 + \frac{[\tilde{x}]_t}{[h]_0} + \frac{\sum_{\kappa=1}^{n}[gx_\kappa c_\kappa^1]_0}{[h]_0}) - ([\tilde{y}^2]_t + \frac{[\tilde{x}]_t}{[h]_0} + \frac{\sum_{\kappa=1}^{n}[gx_\kappa c_\kappa^2]_0}{[h]_0})}{-[H^1]_0 + [H^2]_0}
$$

We can then calculate $[y']_t = \sum_{\kappa=1}^{n}[y_\kappa]_t - \sum_{\kappa=1;\kappa\neq\pi}^{n}[y_\kappa]_t$ since we know $[y]_\kappa$ for all but the target.

We now break the simulation into three cases:

Case 1 $\mathcal{E}([x]'_t, [1]_{\kappa-t}) = [x]_\kappa$, this case $[x']_t$ is a valid answer to the EDP instance and we succeeded.

Case 2 . We have extracted from the adversary a pair $([x']_t, [y']_t)$ which is a valid solution to

the challenge public key $(gx + hy)$ but not the pair $([x]_t, [y]_t)$ which we used to construct it. We now know $([x']_t, [y']_t, [y]_t)$ and wish to find the challenge $[x]_t$, which we calculate as $[x]_t = \frac{[gx']_t + [hy']_t - [hy]_t}{[g]_0}$. We then return $[x]_t$ and succeeded.

Case 3 The adversary has returned a private key for a public key for which we already knew the private key. In this case we fail to reduce.

By the forking lemma, the chance of each successful rewind simulation is at least $\xi/4$, where ξ is the probability that $\mathcal{A}$ successfully forges a signature. Hence, the probability that for a given adversary $\mathcal{A}$ we can extract $[x]_{l-1}$ is $\frac{1}{n} * \frac{1}{t} * \frac{\xi}{4} = \frac{\xi}{4nt}$, where n is the number of queries to the joining oracles and t is the number of time periods. □

5.7 Generalisations

It is easy to use multiple maps, n, together to create $t * n$ total time periods, where t was the number of time periods available from one map. However, more interestingly it appears to be possible to use n elements on the same map to get $\binom{t+2}{n+1}$ time periods, where t was the original time period. We note that for $k = n/2$ the number of time periods is super exponential in the parameter n, however so does the final state size; for this reason we suggest choosing k as a constant which will then allow any polynomial growth in the number of time periods.

It is possible to construct a key tree like that below, where the user also remembers $[\alpha]_1$ and $[\beta]_1$. The user can systemically construct and follow the tree from any given point but an adversary gaining access at any point is not able to generate previous keys without breaking the Set-Exponent Multilinear Decoding Problem ((n,k)-SEMDP).

$t = 0$: $[\alpha^1\beta^0]_0$ $[\alpha^0\beta^1]_0$

$t = 1$: $[\alpha^2\beta^0]_1$ $[\alpha^1\beta^1]_1$ $[\alpha^0\beta^2]_1$

$t = 2$: $[\alpha^3\beta^0]_2$ $[\alpha^2\beta^1]_2$ $[\alpha^1\beta^2]_2$ $[\alpha^0\beta^3]_2$

$t = 3$: $[\alpha^4\beta^0]_3$ $[\alpha^3\beta^1]_3$ $[\alpha^2\beta^2]_3$ $[\alpha^1\beta^3]_3$ $[\alpha^0\beta^4]_3$

The elements $[\alpha]_1$ and $[\beta]_1$ are added to the public key. The public tree then looks like:

$t_0 = l/2 + 0$: $[g\alpha^1\beta^0]_{t_0}$ $[g\alpha^0\beta^1]_{t_0}$

$t_1 = l/1 + 1$: $[g\alpha^2\beta^0]_{t_1}$ $[g\alpha^1\beta^1]_{t_1}$ $[g\alpha^0\beta^2]_{t_1}$

$t_2 = l/2 + 2$: $[g\alpha^3\beta^0]_{t_2}$ $[g\alpha^2\beta^1]_{t_2}$ $[g\alpha^1\beta^2]_{t_2}$ $[g\alpha^0\beta^3]_{t_2}$

$t_3 = l/2 + 3$: $[g\alpha^4\beta^0]_{t_3}$ $[g\alpha^3\beta^1]_{t_3}$ $[g\alpha^2\beta^2]_{t_3}$ $[g\alpha^1\beta^3]_{t_3}$ $[g\alpha^0\beta^4]_{t_3}$

Definition 23 ((n,k)-Set-Exponent Multilinear Decoding Problem ((n,k)-SEMDP)). *Given k generators $\alpha_1, \alpha_2, ..., \alpha_k$ on a map of size greater than $2 * n$, $l \in n$ with a target vector $T = (e_1, e_2, e_3, ..., e_k)$ such that $\sum_{i=1}^{k} e_i = l + 1$,*

$$\mathcal{A}\left\{\begin{array}{l}[\alpha_i]_1, \\ \prod [\alpha_i^{e_i}]_l : \forall(e_i) \in T^c\end{array}\right\} \rightarrow [\prod \alpha_i^{e_i}] : e_i \in T$$

This generalisation of the linkable ring signature, while interesting, reduces to a weaker problem and the final state size grows as a k-root of the number of time periods. It is an interesting area of future work to see how this could be improved.

5.8 Conclusion

We have presented the first candidate strategy for a linkable ring signature scheme with unconditional anonymity and forward security. Such a result would be particularly important for its direct applications to internet voting, as it would resolve a number of heretofore conflicting requirements regarding ease of key registration, voter privacy, unconditional anonymity, and forward security under non-interactive key update. Interestingly our scheme would not require any update to public keys, making it well suited for practical and lightweight protocols. We give formal proofs of all security properties under tight or almost tight reductions to the Discrete Logarithm and Multilinear Decoding problems. In addition we propose an combinatorial leveraging trick that allows the number of time frames to be substantially increased for a given choice of map.

Chapter 6

Conclusions and Recommendations

> *Without the right of privacy, there is no real freedom of speech or freedom of opinion, and so there is no actual democracy.*
>
> (Dilma Rousseff)

Hasty and injudicious implementation of new digital systems into our government elections is a real and present threat. The routine hacks of major organisations we see in every other area of society has the potential to occur just as regularly against the core of our democracies. This will undermine both the validity of elections and the populace's trust in them. Much greater efforts are required to shield our communications, commerce and particularly our elections from these routine attacks. The financial harm, identity theft and social consequences caused by these routine attacks is a serious issue.

Elections are particularly vulnerable to attacks due to the high-stake rewards which attract powerful nation states. We have also seen an unwillingness of some elected officials and electoral officers to take the threat of attack seriously. Instead, the threat has been denied, and attacks and possible attacks concealed.

End to end verifiable electronic voting uses cryptography to provide strong evidence of correctness. It has the potential to allow secure in-polling booth electronic voting. Work, however, is still required to reduce the number of trust assumptions necessary.

Our progress on removing trusted authorities from cryptographic protocols is a step towards the goal of removing reliance on (trusting) the information security capacity of organisations to protect our data.

6.1 Summary of the Research

Our research presents new ways to reduce trust assumptions required in electronic voting, in both the online and in-polling booth environments. This was done through creating new ways to involves humans in cryptography, utilizing new primitives and distributed systems, and presenting new cryptographic primitives.

6.1.1 Truly Multi-Authority *'Prêt-à-Voter'*

We proposed a refinement of Prêt-à-Voter in-polling booth end-to-end verifiable electronic voting schemes which provides privacy against *ad hoc* compromises of individual poll-site devices, without dependence on prior secrets or personal trusted devices. In addition, our improvement alleviates the privacy issues of ballot generation and storage since their contents are no longer required to be secret, greatly simplifying pre-election logistics, while also solving the problem of combining forward secrecy and auditability of printers. Our solution relies on the use of autonomous individual-ballot third-party mixing and re-encryption inside the polling booth.

While the need for additional in-booth hardware may make our approach unsuitable for some voting scenarios, we contend that the benefits outweigh the costs. This is particularly true in situations where the voters are unwilling to trust the election authority, or fear reprisal for voting their conscience—an increasing concern worldwide. The method we present seems inapplicable to STAR-Vote or any scheme which uses direct encryption. It is an issue of ongoing investigation whether Scantegrity II can be adjusted to have re-mixable confirmation codes, which would make it eligible for the privacy enhancement of our technique.

6.1.2 Conceptually Simple Remote Voting against Tiny Tyrants

We presented a practical remote voting scheme called VOTOR which provides security from the election authority and tellers. It uses linkable ring signatures and commodity anonymous channels, such as Tor. We provided an explanation of how to adapt Tor to provide higher levels of privacy for this setting. We presented a framework for evaluating the security of voting schemes which we use to analyse VOTOR. The security of VOTOR compares favourably to popular and recent schemes such as Helios and Remotegrity on many (but not all) aspects, largely due to

the simple reduction to secure cryptographic primitives. In addition, its simplicity and use of freely available existing systems enables high levels of practicality and voter verification which surpasses the other systems. Useability issues are mitigated since individual voters can trust the correctness of the voting software. The software can be audited by independent experts, which provides security in this model without directly impacting usability.

6.1.3 Forward Secure Linkable Ring Signatures

We presented the first candidate strategy for a linkable ring signature scheme with unconditional anonymity and forward security. This result would be particularly important for its direct applications to internet voting, as it would resolve a number of heretofore conflicting requirements regarding ease of key registration, voter privacy, unconditional anonymity, and forward security under a non-interactive key update. Interestingly our scheme would not require any update to public keys, making it well suited for practical and lightweight protocols. We give formal proofs of all security properties under tight or almost tight reductions to the Discrete Logarithm and Multilinear Decoding problems. In addition we propose an combinatorial leveraging trick that allows the number of time frames to be substantially increased for a given choice of map. Unfortunately, all candidate multilinear maps are currently broken and hence this scheme is unrealisable until they are fixed. It is worth noting, however, that many papers continue to be published in the best conferences in the field based on this assumption. Nevertheless, the tight or almost tight reductions apply to Liu et al's non-forward secure bilinear map construction and greatly improve its efficiency and scalability.

6.2 Future Work

The proposed constructions have not been comprehensively analysed for efficiency and useability; this would be an interesting area of future work. In chapters 4 and 5, the efficiency is largely dependent on the linkable ring signature or multilinear map candidate used, respectively. Nevertheless, a comprehensive analysis of this for given election sizes would be informative. Additionally, the constructions presented in both chapters 3 and 4 would benefit from prototyping with usability and acceptance tests.

Chapter 3 would be significantly improved by providing formal proofs of the claimed properties. However, doing this formally with sensible assumptions is left as future work.

Schemes based on pseudo PKs are popular in the voting literature. It is an interesting area of future work to compare linkable ring signature based schemes, such as those presented in chapters 4 and 5, with scheme based on pseudo PKs. All pseudo PK based schemes known to the author are more complicated than VOTOR at a protocol level. In addition, they make use of Zero-Knowledge Proofs of Knowledge which cast their computational efficiency, compared to VOTOR, into doubt.

6.3 Recommendations

We have a series of recommended research directions to improve the practical security of commerce, communications and elections by reducing trust in authorities and devices.

- **Improving Human Involvement In Protocols** to achieve cryptographic goals is a powerful tool which is largely undeveloped. Part of the reason for this is the multidisciplinary work needed. Methods are required which are both cryptographically secure and humanly useable. Such methods can be further broken down into symmetric and asymmetric.
  - **Symmetric** cryptography works using a shared secret between two or more parties. Most human cryptography is symmetric and most is unconditionally (information theoretically) secure. The list of candidates for symmetric human cryptography is long. Two major ongoing research problem are to determine which functions are easiest for humans to calculate, and to understand how best to prime users.
  - **Asymmetric** cryptography relies on one-way functions. It has the potential to resolve the conflict between authority and device independence in the remote setting. Until recently no examples were known, but recent work by Boyen[Boy16a][Boy16b] provides a beginning in this field. More work is needed to unlock the potential of this area.
- **Increasing Decentralisation** is important for security. The strong centralisation of authority which is present in many electronic systems creates a single point of failure. Both Cryptocurrencies and Anonymity Networks are examples of increased decentralisation

in the wider setting. However, even in its online variants, electronic voting still tends towards heavy centralisation of trust.

- **Toughening Security Models** is another prerequisite for more secure voting. Recent work in deniable encryption is a good example of where security models fail to capture reality. The schemes allow the user to send the ciphertext-plaintext pair they wish, but then produce "evidence"[1] that they encrypted a different plaintext. This provides protection after the fact, but an active adversary can provide randomness to the user and check the ciphertext for consistency. Secure electronic voting requires real world models wherever possible; any gap likely introduces an attack.

There is a strong centralisation of trust in the correctness and security of devices and organisations. As it stands, we need these devices and organisations to be secure to protect our commerce, communication and elections. We have, however, repeatedly seen breaches. There is a pressing need to reduce our reliance on these trusted entities, particularly in high stake areas like government elections. The work presented in this thesis is a step in that direction.

[1] In this case the evidence is randomness, which, when put with the coerced message into the encryption function returns the ciphertext.

Literature Cited

[ABD16] Martin R. Albrecht, Shi Bai, and Léo Ducas. A subfield lattice attack on overstretched NTRU assumptions - cryptanalysis of some FHE and graded encoding schemes. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part I*, volume 9814 of *Lecture Notes in Computer Science*, pages 153–178. Springer, 2016.

[Abe99] Masayuki Abe. Mix-networks on permutation networks. In *Advances in cryptology-ASIACRYPT99*, pages 258–273. Springer, 1999.

[Adi06] Ben Adida. *Advances in cryptographic voting systems*. PhD thesis, Massachusetts Institute of Technology, 2006.

[Adi08] Ben Adida. Helios: Web-based open-audit voting. In Paul C. van Oorschot, editor, *USENIX Security Symposium*, pages 335–348. USENIX Association, 2008.

[Adi10] Ben Adida. Helios v3 verification specs. Technical report, Helios Voting, 2010.

[ADMP+09] Ben Adida, Olivier De Marneffe, Olivier Pereira, Jean-Jacques Quisquater, et al. Electing a university president using open-audit voting: Analysis of real-world use of helios. *EVT/WOTE*, 9:10–10, 2009.

[ALBD04] Riza Aditya, Byoungcheon Lee, Colin Boyd, and Ed Dawson. An efficient mixnet-based voting scheme providing receipt-freeness. In *Trust and Privacy in Digital Business*. Springer, 2004.

[AMV15] Giuseppe Ateniese, Bernardo Magri, and Daniele Venturi. Subversion-resilient signature schemes. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 364–375. ACM, 2015.

[Arr50] Kenneth J Arrow. A difficulty in the concept of social welfare. *Journal of political economy*, 58(4):328–346, 1950.

[BBK+12] Josh Benaloh, Mike Byrne, Philip T. Kortum, Neal McBurnett, Olivier Pereira, Philip B. Stark, and Dan S. Wallach. Star-vote: A secure, transparent, auditable, and reliable voting system. *CoRR*, abs/1211.1904, 2012.

[BBSU12] Simon Barber, Xavier Boyen, Elaine Shi, and Ersin Uzun. Bitter to better - how to make bitcoin a better currency. In Angelos D. Keromytis, editor, *Financial Cryptography*, volume 7397 of *Lecture Notes in Computer Science*, pages 399–414. Springer, 2012.

[BCH+12a] Craig Burton, Chris Culnane, James Heather, Thea Peacock, Peter Y. A. Ryan, Steve Schneider, Sriramkrishnan Srinivasan, Vanessa Teague, Roland Wen, and Zhe Xia. A supervised verifiable voting protocol for the Victorian Electoral Commission. In Manuel J. Kripp, Melanie Volkamer, and Rüdiger Grimm, editors, *Electronic Voting*, volume 205 of *LNI*, pages 81–94. GI, 2012.

[BCH+12b] Craig Burton, Chris Culnane, James Heather, Thea Peacock, Peter YA Ryan, Steve Schneider, Sriramkrishnan Srinivasan, Vanessa Teague, Roland Wen, and Zhe Xia. Using prêt à voter in Victorian state elections. In *Proc. USENIX EVT/WoTE*, 2012.

[BCH+12c] Craig Burton, Chris Culnane, James Heather, Thea Peacock, Peter YA Ryan, Steve Schneider, Sriramkrishnan Srinivasan, Vanessa Teague, Roland Wen, and Zhe Xia. Using prêt a voter in Victorian state elections. In *Proceedings of the 2012 international conference on Electronic Voting Technology/Workshop on Trustworthy Elections*, pages 1–1. USENIX Association, 2012.

[BCP+11] David Bernhard, Véronique Cortier, Olivier Pereira, Ben Smyth, and Bogdan Warinschi. Adapting helios for provable ballot privacy. In Vijay Atluri and Claudia Díaz, editors, *Computer Security - ESORICS 2011 - 16th European Symposium on Research in Computer Security, Leuven, Belgium, September 12-14, 2011. Proceedings*, volume 6879 of *Lecture Notes in Computer Science*, pages 335–354. Springer, 2011.

[BCPW12] David Bernhard, Véronique Cortier, Olivier Pereira, and Bogdan Warinschi. Measuring vote privacy, revisited. In Ting Yu, George Danezis, and Virgil D. Gligor, editors, *the ACM Conference on Computer and Communications Security, CCS'12, Raleigh, NC, USA, October 16-18, 2012*, pages 941–952. ACM, 2012.

[Ben87] Josh Benaloh. *Verifiable secret-ballot elections*. PhD thesis, Yale University, 1987.

[Ben13] Josh Benaloh. Rethinking voter coercion: The realities imposed by technology. In *2013 Electronic Voting Technology Workshop / Workshop on Trustworthy Elections, EVT/WOTE '13, Washington, D.C., USA, August 12-13, 2013*. USENIX Association, 2013.

[BGI+12] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. *J. ACM*, 59(2):6:1–6:48, 2012.

[BH15] Mihir Bellare and Viet Tung Hoang. Resisting randomness subversion: Fast deterministic and hedged public-key encryption in the standard model. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 627–656. Springer, 2015.

[BLS+03] Benjamin B. Bederson, Bongshin Lee, Robert M. Sherman, Paul S. Herrnson, and Richard G. Niemi. Electronic voting system usability issues. In Gilbert Cockton and Panu Korhonen, editors, *CHI*, pages 145–152. ACM, 2003.

[Boy90] Colin Boyd. A new multiple key cipher and an improved voting scheme. In *Advances in CryptologyEUROCRYPT89*, pages 617–625. Springer, 1990.

[Boy16a] Xavier Boyen. The case for human primacy in cryptography. MyCRYPT 2016: The International Conference on Cryptology & Malicious Security hosted in Malaysia, 2016.

[Boy16b] Xavier Boyen. Humanly decipherable public-key encryption: Yes, really! ASIACrypt 2016: 22nd Annual International Conference on the Theory and Applications of Cryptology and Information Security, 2016.

[BPR14] Mihir Bellare, Kenneth G. Paterson, and Phillip Rogaway. Security of symmetric encryption against mass surveillance. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part I*, volume 8616 of *Lecture Notes in Computer Science*, pages 1–19. Springer, 2014.

[BPW12] David Bernhard, Olivier Pereira, and Bogdan Warinschi. How not to prove yourself: Pitfalls of the fiat-shamir heuristic and applications to helios. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 626–643. Springer, 2012.

[Bru66] Robert Sherrick Brumbaugh. *Ancient Greek Gadgets and Machines*. Crowell, 1966.

[BS03] Dan Boneh and Alice Silverberg. Applications of multilinear forms to cryptography. *Contemporary Mathematics*, 324(1):71–90, 2003.

[BT94] Josh Cohen Benaloh and Dwight Tuinstra. Receipt-free secret-ballot elections (extended abstract). In Frank Thomson Leighton and Michael T. Goodrich, editors, *STOC*, pages 544–553. ACM, 1994.

[BWZ14] Dan Boneh, David J Wu, and Joe Zimmerman. Immunizing multilinear maps against zeroizing attacks. *IACR Cryptology ePrint Archive*, 2014:930, 2014.

[BY86] Josh C Benaloh and Moti Yung. Distributing the power of a government to enhance the privacy of voters. In *Proceedings of the fifth annual ACM symposium on Principles of distributed computing*, pages 52–62. ACM, 1986.

[CC07] Malcolm Crook and Tom Crook. The advent of the secret ballot in britain and france, 1789–1914: From public assembly to private compartment. *History*, 92(308):449–471, 2007.

[CCaJC+10] Richard Carback, David Chaum, Jeremy Clark abd John Conwaym, Aleksander Essex, Paul S. Hernson, 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. In *Proc. USENIX Accurate Electronic Voting Technology Workshop*, 2010.

[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 codes. In *EVT*. USENIX Association, 2008.

[CCM08a] Michael R Clarkson, Stephen Chong, and Andrew C Myers. Civitas: Toward a secure voting system. In *Proceedings of the 2008 IEEE Symposium on Security and Privacy*, pages 354–368. IEEE Computer Society, 2008.

[CCM08b] Michael R Clarkson, Stephen Chong, and Andrew C Myers. Civitas: Toward a secure voting system. In *Proc. IEEE Symposium on Security and Privacy*, 2008.

[CDNO97] Ran Canetti, Cynthia Dwork, Moni Naor, and Rafail Ostrovsky. Deniable encryption. In Burton S. Kaliski Jr., editor, *CRYPTO*, volume 1294 of *Lecture Notes in Computer Science*, pages 90–104. Springer, 1997.

[CDPR16] Ronald Cramer, Léo Ducas, Chris Peikert, and Oded Regev. Recovering short generators of principal ideals in cyclotomic rings. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II*, volume 9666 of *Lecture Notes in Computer Science*, pages 559–585. Springer, 2016.

[CF85a] Josh D Cohen and Michael J Fischer. A robust and verifiable cryptographically secure election scheme. In *FOCS*, volume 85, pages 372–382, 1985.

[CF85b] Josh D Cohen and Michael J Fischer. A robust and verifiable cryptographically secure election scheme. In *FOCS*, pages 372–382, 1985.

[CFSY96] Ronald Cramer, Matthew Franklin, Berry Schoenmakers, and Moti Yung. Multi-authority secret-ballot elections with linear work. In *EUROCRYPT*, 1996.

[CGS97] Ronald Cramer, Rosario Gennaro, and Berry Schoenmakers. A secure and optimally efficient multi-authority election scheme. *European Transactions on Telecommunications*, 8(5):481–490, 1997.

[Cha81] David Chaum. Untraceable mail, return addresses and digital pseudonyms. *Communications of the ACM*, 24(2):84–88, 1981.

[Cha82] David Chaum. Blind signatures for untraceable payments. In *CRYPTO*, 1982.

[Cha85] David Chaum. Security without identification: Transaction systems to make big brother obsolete. *Communications of the ACM*, 28(10):1030–1044, 1985.

[Cha88a] David Chaum. The dining cryptographers problem: Unconditional sender and recipient untraceability. *Journal of cryptology*, 1(1):65–75, 1988.

[Cha88b] David Chaum. Elections with unconditionally-secret ballots and disruption equivalent to breaking rsa. In Christoph G. Günther, editor, *EUROCRYPT*, volume 330 of *Lecture Notes in Computer Science*, pages 177–182. Springer, 1988.

[Cha88c] David Chaum. Elections with unconditionally-secret ballots and disruption equivalent to breaking rsa. In *Advances in CryptologyEUROCRYPT88*, pages 177–182. Springer, 1988.

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

[CHJ+13] Chris Culnane, James Heather, Rui Joaquim, Peter Y A Ryan, Steve Schneider, and Vanessa Teague. Faster print on demand for prêt à voter. *USENIX J. Election Technology and Systems*, 2013.

[CHL+15] Jung Hee Cheon, Kyoohyung Han, Changmin Lee, Hansol Ryu, and Damien Stehlé. Cryptanalysis of the multilinear map over the integers. In *Advances in Cryptology–EUROCRYPT 2015*, pages 3–12. Springer, 2015.

[CJL16] Jung Hee Cheon, Jinhyuck Jeong, and Changmin Lee. An algorithm for ntru problems and cryptanalysis of the ggh multilinear map without a low level encoding of zero. *IACR Cryptology ePrint Archive*, 2016.

[CLT13] Jean-Sébastien Coron, Tancrede Lepoint, and Mehdi Tibouchi. Practical multilinear maps over the integers. In *Advances in Cryptology–CRYPTO 2013*, pages 476–493. Springer, 2013.

[Coh86] Josh D Cohen. *Improving privacy in cryptographic elections*. Citeseer, 1986.

[CRS05] David Chaum, Peter Y. A. Ryan, and Steve A. Schneider. A practical voter-verifiable election scheme. In *ESORICS*, 2005.

[CRST15a] Chris Culnane, Peter Y. A. Ryan, Steve A. Schneider, and Vanessa Teague. vvote: A verifiable voting system. *ACM Trans. Inf. Syst. Secur.*, 18(1):3:1–3:30, 2015.

[CRST15b] Chris Culnane, Peter Y. A. Ryan, Steve A. Schneider, and Vanessa Teague. vVote: A verifiable voting system. *ACM Trans. Inf. Syst. Secur.*, 18(1):3, 2015.

[CS13] Véronique Cortier and Ben Smyth. Attacking and fixing helios: An analysis of ballot secrecy. *Journal of Computer Security*, 21(1):89–148, 2013.

[CS14] Chris Culnane and Steve A. Schneider. A peered bulletin board for robust use in verifiable voting systems. In *IEEE 27th Computer Security Foundations Symposium, CSF 2014, Vienna, Austria, 19-22 July, 2014*, pages 169–183. IEEE Computer Society, 2014.

[CSST06] Sébastien Canard, Berry Schoenmakers, Martijn Stam, and Jacques Traoré. List signature schemes. *Discrete Applied Mathematics*, 154(2):189–201, 2006.

[CvH91] David Chaum and Eugène van Heyst. Group signatures. In Donald W. Davies, editor, *EUROCRYPT*, volume 547 of *Lecture Notes in Computer Science*, pages 257–265. Springer, 1991.

[DGG+15] Yevgeniy Dodis, Chaya Ganesh, Alexander Golovnev, Ari Juels, and Thomas Ristenpart. A formal treatment of backdoored pseudorandom generators. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 101–126. Springer, 2015.

[DKR06] Stephanie Delaune, Steve Kremer, and Mark Ryan. Coercion-resistance and receipt-freeness in electronic voting. In *Computer Security Foundations Workshop, 2006. 19th IEEE*, pages 12–pp. IEEE, 2006.

[DVDGA12] Denise Demirel, Jeroen Van De Graaf, and Roberto Araújo. Improving helios with everlasting privacy towards the public. In *Proceedings of the 2012 international conference on Electronic Voting Technology/Workshop on Trustworthy Elections*, pages 8–8. USENIX Association, 2012.

[ECHA09a] Aleks Essex, Jeremy Clark, Urs Hengartner, and Carlisle Adams. How to print a secret. In *Proc. USENIX Hot Topics in Security*, 2009.

[ECHA09b] Aleks Essex, Jeremy Clark, Urs Hengartner, and Carlisle Adams. How to print a secret. In *Proc. USENIX Hot Topics in Security*, 2009.

[FOO93] Atsushi Fujioka, Tatsuaki Okamoto, and Kazuo Ohta. A practical secret voting scheme for large scale elections. In *Advances in CryptologyAUSCRYPT'92*, pages 244–251. Springer, 1993.

[FS07] Eiichiro Fujisaki and Koutarou Suzuki. Traceable ring signature. In *Public Key Cryptography–PKC 2007*, pages 181–200. Springer, 2007.

[Fuj12] Eiichiro Fujisaki. Sub-linear size traceable ring signatures without random oracles. *IEICE TRANSACTIONS on Fundamentals of Electronics, Communications and Computer Sciences*, 95(1):151–166, 2012.

[GGH13] Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices. In *Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings*, pages 1–17, 2013.

[GGH15] Craig Gentry, Sergey Gorbunov, and Shai Halevi. Graph-induced multilinear maps from lattices. In *Theory of Cryptography*, pages 498–527. Springer, 2015.

[GGOR14] Juan A. Garay, Clinton Givens, Rafail Ostrovsky, and Pavel Raykov. Fast and unconditionally secure anonymous channel. In Magnús M. Halldórsson and Shlomi Dolev, editors, *ACM Symposium on Principles of Distributed Computing, PODC '14, Paris, France, July 15-18, 2014*, pages 313–321. ACM, 2014.

[GKK+06] Marcin Gogolewski, Marek Klonowski, Przemyslaw Kubiak, Miroslaw Kutylowski, Anna Lauks, and Filip Zagórski. Kleptographic attacks on e-voting schemes. In *ETRICS*, 2006.

[GMR85] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof-systems (extended abstract). In Robert Sedgewick, editor, *Proceedings of the 17th Annual ACM Symposium on Theory of Computing, May 6-8, 1985, Providence, Rhode Island, USA*, pages 291–304. ACM, 1985.

[Gru12] Eitan Grundland. An analysis of the wombat voting system model, 2012.

[HB16a] Thomas Haines and Xavier Boyen. Truly multi-authority prêt-à-voter. In *International Joint Conference on Electronic Voting*, pages 56–72. Springer, 2016.

[HB16b] Thomas Haines and Xavier Boyen. Votor: conceptually simple remote voting against tiny tyrants. In *Proceedings of the Australasian Computer Science Week Multiconference*, page 32. ACM, 2016.

[HJ16] Yupu Hu and Huiwen Jia. Cryptanalysis of ggh map. In *Advances in Cryptology - EUROCRYPT 2016, Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 537–565. Springer, 2016.

[HL07] Chengyu Hu and Daxing Li. Forward-secure traceable ring signature. In Wenying Feng and Feng Gao, editors, *SNPD (3)*, pages 200–204. IEEE Computer Society, 2007.

[HS00a] Martin Hirt and Kazue Sako. Efficient receipt-free voting based on homomorphic encryption. In *Advances in CryptologyEUROCRYPT 2000*, pages 539–556. Springer, 2000.

[HS00b] Martin Hirt and Kazue Sako. Efficient receipt-free voting based on homomorphic encryption. In *EUROCRYPT*, 2000.

[HST14] James Heather, Steve Schneider, and Vanessa Teague. Cryptographic protocols with everyday objects. *Formal Asp. Comput.*, 26(1):37–62, 2014.

[JCJ05] Ari Juels, Dario Catalano, and Markus Jakobsson. Coercion-resistant electronic elections. In *Proc. WPES*, 2005.

[KK98] Michael S Kochin and Levis A Kochin. When is buying votes wrong? *Public Choice*, 97(4):645–662, 1998.

[KKLZ05] Marek Klonowski, Mirosław Kutyłowski, Anna Lauks, and Filip Zagórski. A practical voting scheme with receipts. In *Information Security*, pages 490–497. Springer, 2005.

[KT09] Ralf Küsters and Tomasz Truderung. An epistemic approach to coercion-resistance for electronic voting protocols. In *Security and Privacy, 2009 30th IEEE Symposium on*, pages 251–266. IEEE, 2009.

[LASZ14] Joseph K. Liu, Man Ho Au, Willy Susilo, and Jianying Zhou. Linkable ring signature with unconditional anonymity. *IEEE Transactions on Knowledge and Data Engineering*, 26(1):157–165, 2014.

[LBD+03] Byoungcheon Lee, Colin Boyd, Ed Dawson, Kwangjo Kim, Jeongmo Yang, and Seungjae Yoo. Providing receipt-freeness in mixnet-based voting protocols. In *Info. Sec. and Cryptol.–ICISC*. Springer, 2003.

[LSS14] Adeline Langlois, Damien Stehlé, and Ron Steinfeld. Gghlite: More efficient multilinear maps from ideal lattices. In *Advances in Cryptology–EUROCRYPT 2014*, pages 239–256. Springer, 2014.

[LW05] JosephK. Liu and DuncanS. Wong. Linkable ring signatures: Security models and new schemes. In Osvaldo Gervasi, MarinaL. Gavrilova, Vipin Kumar, Antonio Lagan, HeowPueh Lee, Youngsong Mun, David Taniar, and ChihJengKenneth Tan, editors, *Computational Science and Its Applications ICCSA 2005*, volume 3481 of *Lecture Notes in Computer Science*, pages 614–623. Springer Berlin Heidelberg, 2005.

[LWW04] Joseph K Liu, Victor K Wei, and Duncan S Wong. Linkable spontaneous anonymous group signature for ad hoc groups. In *Information Security and Privacy*, pages 325–335. Springer, 2004.

[Mer83] M Merrit. Cryptographic protocols. Ph.D. Thesis, 1983.

[MN10] Tal Moran and Moni Naor. Split-ballot voting: Everlasting privacy with distributed trust. *ACM Trans. Inf. Syst. Secur.*, 13(2), 2010.

[Mol06] Richard A Mollin. *An introduction to cryptography*. CRC Press, 2006.

[MRV99] Silvio Micali, Michael O. Rabin, and Salil P. Vadhan. Verifiable random functions. In *40th Annual Symposium on Foundations of Computer Science, FOCS '99, 17-18 October, 1999, New York, NY, USA*, pages 120–130. IEEE Computer Society, 1999.

[MSZ16] Eric Miles, Amit Sahai, and Mark Zhandry. Annihilation attacks for multilinear maps: Cryptanalysis of indistinguishability obfuscation over ggh13. In *Annual Cryptology Conference*, pages 629–658. Springer, 2016.

[Nak08] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2008.

[Nef01] C. Andrew Neff. A verifiable secret shuffle and its application to e-voting. In *CCS*, 2001.

[Nef04] C Andrew Neff. Practical high certainty intent verification for encrypted votes. *VoteHere document*, 2004.

[Oka96] Tatsuaki Okamoto. An electronic voting scheme. In *Advanced IT Tools*, pages 21–30. Springer, 1996.

[Oka98a] Tatsuaki Okamoto. Receipt-free electronic voting schemes for large scale elections. In *Security Protocols*, pages 25–35. Springer, 1998.

[Oka98b] Tatsuaki Okamoto. Receipt-free electronic voting schemes for large scale elections. In *Security Protocols*, pages 25–35. Springer, 1998.

[Ped91] Torben P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In Joan Feigenbaum, editor, *Advances in Cryptology - CRYPTO '91, 11th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1991, Proceedings*, volume 576 of *Lecture Notes in Computer Science*, pages 129–140. Springer, 1991.

[Pfi95] Birgit Pfitzmann. Breaking an efficient anonymous channel. In *Advances in CryptologyEUROCRYPT'94*, pages 332–340. Springer, 1995.

[PIK93] Choonsik Park, Kazutomo Itoh, and Kaoru Kurosawa. Efficient anonymous channel and all/nothing election scheme. In *EUROCRYPT*, 1993.

[PS96a] David Pointcheval and Jacques Stern. Security proofs for signature schemes. In Ueli M. Maurer, editor, *Advances in Cryptology - EUROCRYPT '96, International Conference on the Theory and Application of Cryptographic Techniques, Saragossa, Spain, May 12-16, 1996, Proceeding*, volume 1070 of *Lecture Notes in Computer Science*, pages 387–398. Springer, 1996.

[PS96b] David Pointcheval and Jacques Stern. Security proofs for signature schemes. In *Eurocrypt*, volume 96, pages 387–398. Springer, 1996.

[PS15] Omer Paneth and Amit Sahai. On the equivalence of obfuscation and multilinear maps. *IACR Cryptology ePrint Archive*, 2015:791, 2015.

[Rad13] Kenneth J Radke. Security ceremonies: including humans in cryptographic protocols, 2013.

[Rez10] PedroA.D. Rezende. Electronic elections: A balancing act. In David Chaum, Markus Jakobsson, RonaldL. Rivest, PeterY.A. Ryan, Josh Benaloh, Miroslaw Kutylowski, and Ben Adida, editors, *Towards Trustworthy Elections*, volume 6000 of *Lecture Notes in Computer Science*, pages 124–140. Springer Berlin Heidelberg, 2010.

[Riv06] Ronald L. Rivest. The threeballot voting system. *Unpublished draft*, 2006.

[RST01a] Ronald L Rivest, Adi Shamir, and Yael Tauman. How to leak a secret. In *Advances in CryptologyASIACRYPT 2001*, pages 552–565. Springer, 2001.

[RST01b] Ronald L. Rivest, Adi Shamir, and Yael Tauman. How to leak a secret. In Colin Boyd, editor, *Advances in Cryptology - ASIACRYPT 2001, 7th International Conference on the Theory and Application of Cryptology and Information Security, Gold Coast, Australia, December 9-13, 2001, Proceedings*, volume 2248 of *Lecture Notes in Computer Science*, pages 552–565. Springer, 2001.

[RST15] Peter Y. A. Ryan, Steve A. Schneider, and Vanessa Teague. End-to-end verifiability in voting systems, from theory to practice. *IEEE Security & Privacy*, 13(3):59–62, 2015.

[RT09] Peter Y A Ryan and Vanessa Teague. Ballot permutations in pret a voter. In *Proc. Electronic Voting Technology/Workshop on Trustworthy Elections*, 2009.

[Rya05] P.Y.A Ryan. A variant of the Chaum voter-verifiable scheme. In *Proceedings of the 2005 workshop on Issues in the theory of security*, pages 81–88. ACM, 2005.

[Rya10] PeterY.A. Ryan. The computer ate my vote. In *Formal Methods: State of the Art and New Directions*. Springer, 2010.

[SB13] Ben Smyth and David Bernhard. Ballot secrecy and ballot independence coincide. In Jason Crampton, Sushil Jajodia, and Keith Mayes, editors, *Computer Security - ESORICS 2013 - 18th European Symposium on Research in Computer Security, Egham, UK, September 9-13, 2013. Proceedings*, volume 8134 of *Lecture Notes in Computer Science*, pages 463–480. Springer, 2013.

[SK95] Kazue Sako and Joe Kilian. Receipt-free mix-type voting scheme. In *EUROCRYPT*, pages 393–403. Springer, 1995.

[SP06] Krishna Sampigethaya and Radha Poovendran. A survey on mix networks and their secure applications. *Proceedings of the IEEE*, 94(12):2142–2181, 2006.

[TPLT13] Georgios Tsoukalas, Kostas Papadimitriou, Panos Louridas, and Panayiotis Tsanakas. From helios to zeus. *Presented as part of the USENIX Journal of Election and Technology and Systems (JETS)*, pages 1–17, 2013.

[Wik04] Douglas Wikström. A universally composable mix-net. In *TCC*, pages 317–335, 2004.

[Yao82] Andrew Chi-Chih Yao. Protocols for secure computations (extended abstract). In *FOCS*, pages 160–164. IEEE Computer Society, 1982.

[YY96] Adam Young and Moti Yung. The dark side of black-box cryptography or: Should we trust capstone? In *Advances in CryptologyCRYPTO96*, pages 89–103. Springer, 1996.

[ZCC+13] Filip Zagórski, Richard Carback, David Chaum, Jeremy Clark, Aleksander Essex, and Poorvi L. Vora. Remotegrity: Design and use of an end-to-end verifiable remote voting system. *IACR Cryptology ePrint Archive*, 2013:214, 2013.

[Zha14] Mark Zhandry. Adaptively secure broadcast encryption with small system parameters. *IACR Cryptology ePrint Archive*, 2014:757, 2014.