End-to-end verifiable voting
with Prêt à Voter

David Bismark

Dissertation for the degree of Doctor of Philosophy
Computing

University of Surrey
Department of Computing
Declaration

This thesis and the work to which it refers are the results of my own effort. Any ideas, data, images or text resulting from the work of others (whether published or unpublished) are fully identified as such within the work and attributed to their originator in text, bibliography or in footnotes. This thesis has not been submitted in whole or in part for any other academic degree or professional qualification. I agree that the University has the right to submit my work to the plagiarism detection service TurnitinUK for originality checks. Whether or not drafts have been so-assessed, the University reserves the right to require an electronic version of the final document (as submitted) for assessment as above.

David Bismark
October 2010
To
Sophie, my beloved wife.
And Lotus, the cat responsible for
all the topos.
End-to-end verifiable voting with Prêt à Voter

David Bismark
University of Surrey
UK
d.bismark@surrey.ac.uk
20th October 2010

Abstract

Transparent and verifiable elections can be achieved by end-to-end verifiable electronic voting systems. The purpose of such systems is to enable voters to verify the inclusion of their vote in the final tally while keeping the votes secret. Achieving verifiability and secrecy at the same time is hard and this thesis explores the properties of verifiable electronic voting systems and describes a set of developments to the end-to-end verifiable electronic voting system Prêt à Voter to achieve these.
Acknowledgements

I would like to thank everyone at the University of Surrey for a wonderful few years — I had a great time from starting as an undergraduate to now finally managing to leave. First of all thanks to my [insert adjective here before printing thesis] supervisor Dr James Heather who, although far too similar in personality and work ethic to myself, has a hands-off approach to leading that somehow works with my own unwillingness to be led and utter inability to accomplish anything but the unexpected. Thanks to Professor Steve Schneider, especially if it was you who made my doctoral training grant go so far. Thank you Dr Helen Treharne who showed a thorough interest in electronic voting while supervising my MSc project, without that none of this would have happened. Thank you Professor Peter Y. A. Ryan who, although not based at Surrey, has been part of all our innovation and most of my papers would not have existed without your help. Many thanks to Zhe Xia for interesting discussions and access to your vast knowledge. Thank you Dr Vanessa Teague for being one of my examiners and spotting that contrary to public belief, Australia was not around in 1858. Thanks also to Dr Roger Peel for various things, including an encouraging speech that ensured that this thesis got in on time.

Thank you Chris Culnane who, adamant he would never do an MSc and then adamant he would never do a PhD, has been the greatest of friends throughout our time together in Guildford. You were adamant you would never be an RA but at least you’re now an RA on an electronic voting project.

My heartfelt thanks to my beautiful wife Sophie Bismark for encouragement and for giving me her surname (my surname was previously Lundin) so that I could move up in the list of authors. Per unit of time this may well be the most efficient way to advance an academic career and I encourage anyone to give it a go.
Further acknowledgements

I confirm that the work presented in this thesis is my own, with the following exceptions:

- In Chapter 3 *An implementation of Prêt à Voter* the work with the implementation was undertaken in a group consisting of James Heather, Roger Peel, Zhe Xia, Phil Howard and myself. My code contribution was specifically in the back-end as I programmed the web bulletin board, tellers, audit machine etc. The writing up of our findings in this chapter is mine, but I am very thankful to the whole group that I could be a part of this excellent project. Our experiences from this project has been published as [13].

- Chapter 6 *A human readable paper audit trail* originated in an idea I had based on Peter Ryan’s existing human-readable paper audit trail [79]. At the time I had this idea, Peter and I were both at Schloss Dagstuhl, that wonderful, creative place. My idea was mainly regarding the linking of the two pages of the ballot form and the auditing properties this resulted in and I am very thankful to Peter for graciously lending the scheme his distributed pre-creation of the onions as well as the cryptographic detail of the scheme. The scheme detailed in this chapter has been published as [54].

- Chapter 7 *Remote voting using paper-based schemes* has its origin in an idea that Stefan Popoveniuc had and that we also discussed at Dagstuhl. Our thoughts on the subject went through a number of iterations and was eventually published as [69]. It is now hard to say exactly where the details come from so I would like to thank Stefan very much for developing these ideas with me.
Contents

2.2.5 Threats ........................................... 23
   The coercer ........................................ 24
   The trusted party .................................. 24
   The malicious voter ................................. 25

2.3 Cryptography ...................................... 25
   2.3.1 RSA ........................................ 26
   2.3.2 Elgamal ...................................... 26
      Generating a key pair ............................. 27
      Encryption ...................................... 27
      Decryption ...................................... 27
      Re-encryption ................................... 28
      Threshold Elgamal ................................ 28
   2.3.3 Paillier ....................................... 29
   2.3.4 Digital signatures ............................ 29
   2.3.5 Threshold cryptography ....................... 29
   2.3.6 Mix networks ................................ 30
      Decryption mix networks ......................... 30
      Re-encryption mix networks ...................... 32
      Random partial checks (RPC) ..................... 33
   2.3.7 Zero-knowledge proofs ......................... 33

2.4 Paper based electronic voting schemes .......... 34
   2.4.1 Chaum’s visual cryptography system ........ 34
   2.4.2 Prêt à Voter ................................ 35
      The ballot form .................................. 35
      Voting ceremony and verifying the vote ......... 36
   2.4.3 Punchscan .................................... 39
      The ballot form .................................. 39
      Voting ceremony ................................ 40
      Meeting of officials and the diskless workstation 40
      The mixing scheme and its initial audit ........ 43
      The decryption of the votes ...................... 45
   2.4.4 Scantegrity .................................. 46
      The ballot form .................................. 47
      The voting ceremony and voter verifiability .... 48
      Challenging the election ......................... 48
      Discussion ..................................... 49
   2.4.5 Scantegrity II ................................. 49
      The ballot form .................................. 50
      The voting ceremony ............................. 50
      Publishing auditing information ................ 50
      Auditing a vote and resolving disputes .......... 51
      Discussion ..................................... 51
2.4.6 Scratch & Vote ........................................... 51
   The ballot form ........................................... 51
   The onion .................................................. 52
   The voting ceremony ..................................... 52
   Auditing a ballot form ................................... 53
   Tabulation .................................................. 53
   Discussion .................................................. 54
2.4.7 Problems in paper-based electronic voting schemes ............ 54
   Authority knowledge ....................................... 54
   Chain of custody ........................................... 55
   Chain voting ............................................... 55
   Randomisation attack ...................................... 56
   Denial of service attack ................................... 57
   The Italian attack ......................................... 57
   Everlasting privacy ....................................... 58

3 An implementation of Prêt à Voter .................................. 61
   3.1 Introduction ............................................ 61
   3.2 Requirements .......................................... 62
   3.3 The system .............................................. 63
      3.3.1 The Java classes .................................... 63
      3.3.2 Networking .......................................... 64
      3.3.3 Teller ............................................... 66
      3.3.4 Web bulletin board .................................. 68
      3.3.5 Registry ............................................ 69
      3.3.6 Booth ................................................ 70
      3.3.7 Audit Machine ....................................... 70
      3.3.8 Website ............................................. 71
   3.4 Testing and trials ....................................... 72
      3.4.1 Newcastle trial 2008 ................................. 72
      The election ............................................. 72
      Room layout ............................................. 72
      Issues .................................................... 72
      Technical issues ......................................... 73
      Lessons ................................................... 73
      Summary .................................................. 74
   3.5 Problems ................................................ 75
      3.5.1 Single Transferable Vote ............................ 75
      3.5.2 Four concurrent polling stations ..................... 76
      3.5.3 Full permutations rather than cyclic shifts ............ 76
      3.5.4 Paper trail .......................................... 77
      3.5.5 Allowing the vote to be spoilt ....................... 77
      3.5.6 More than one race on a ballot paper ................ 78
## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.5.7 Partition according to number of choices</td>
<td>78</td>
</tr>
<tr>
<td>3.6 Summary</td>
<td>79</td>
</tr>
<tr>
<td>3.7 Contribution</td>
<td>79</td>
</tr>
<tr>
<td>3.8 Future directions</td>
<td>80</td>
</tr>
<tr>
<td>4 Component based electronic voting systems</td>
<td>81</td>
</tr>
<tr>
<td>4.1 Introduction</td>
<td>81</td>
</tr>
<tr>
<td>4.1.1 Domains captured</td>
<td>81</td>
</tr>
<tr>
<td>4.1.2 Cascading changes</td>
<td>82</td>
</tr>
<tr>
<td>4.1.3 Discussion on the reach of the proposed methodology</td>
<td>82</td>
</tr>
<tr>
<td>4.2 Component hierarchy</td>
<td>82</td>
</tr>
<tr>
<td>4.2.1 Physical layer</td>
<td>83</td>
</tr>
<tr>
<td>Hardware authentication method</td>
<td>84</td>
</tr>
<tr>
<td>Publishing strategy</td>
<td>84</td>
</tr>
<tr>
<td>Transfer method</td>
<td>84</td>
</tr>
<tr>
<td>4.2.2 Computational layer</td>
<td>84</td>
</tr>
<tr>
<td>Cryptography scheme</td>
<td>84</td>
</tr>
<tr>
<td>Anonymisation strategy</td>
<td>84</td>
</tr>
<tr>
<td>Tallying procedure</td>
<td>85</td>
</tr>
<tr>
<td>4.2.3 Election layer</td>
<td>85</td>
</tr>
<tr>
<td>Election method</td>
<td>85</td>
</tr>
<tr>
<td>Election management system</td>
<td>85</td>
</tr>
<tr>
<td>Voter-ballot box communication channel</td>
<td>86</td>
</tr>
<tr>
<td>4.2.4 Human layer</td>
<td>86</td>
</tr>
<tr>
<td>Voter registration</td>
<td>86</td>
</tr>
<tr>
<td>Ballot form configuration</td>
<td>86</td>
</tr>
<tr>
<td>Polling station layout and management</td>
<td>86</td>
</tr>
<tr>
<td>Verifiability front-end</td>
<td>87</td>
</tr>
<tr>
<td>4.2.5 Component interaction</td>
<td>87</td>
</tr>
<tr>
<td>4.3 Component properties</td>
<td>87</td>
</tr>
<tr>
<td>4.3.1 Layer and slot into which it fits</td>
<td>87</td>
</tr>
<tr>
<td>4.3.2 Origin, requirements and constraints posed by the slot</td>
<td>88</td>
</tr>
<tr>
<td>4.3.3 Verification strategy</td>
<td>88</td>
</tr>
<tr>
<td>4.3.4 Location in the system (authority-close or voter-close)</td>
<td>89</td>
</tr>
<tr>
<td>4.3.5 Threat analysis</td>
<td>89</td>
</tr>
<tr>
<td>4.3.6 Computational complexity analysis</td>
<td>89</td>
</tr>
<tr>
<td>4.4 Examples</td>
<td>90</td>
</tr>
<tr>
<td>4.4.1 Punchscan: an example decomposition</td>
<td>90</td>
</tr>
<tr>
<td>Human layer</td>
<td>90</td>
</tr>
<tr>
<td>Election layer</td>
<td>90</td>
</tr>
<tr>
<td>Computational layer</td>
<td>91</td>
</tr>
<tr>
<td>Physical layer</td>
<td>91</td>
</tr>
<tr>
<td>Component relationships</td>
<td>91</td>
</tr>
<tr>
<td>4.4.2 Changes to cryptography component in Prêt à Voter</td>
<td>92</td>
</tr>
</tbody>
</table>
6.2 The scheme ......................................................... 118
  6.2.1 The ballot form and its use .............................. 118
  6.2.2 Cut-and-choose ............................................. 120
  6.2.3 Generation of the encrypted ballot forms ........... 121
  6.2.4 Anonymising Tabulation .................................. 122
  6.2.5 Audit of the paper trail ................................. 124
6.3 Analysis .......................................................... 125
  6.3.1 Linking the receipt onions to the candidate lists .... 125
  6.3.2 Voter choices differ between pages .................... 126
6.4 Summary .......................................................... 126

7 Remote voting using paper-based schemes 129
  7.1 Introduction .................................................... 129
    7.1.1 Properties of the scheme ............................... 130
    7.1.2 Threat model ............................................. 131
  7.2 Techniques that do not work ............................... 132
    7.2.1 Punchscan ............................................... 132
    7.2.2 Prêt à Voter .............................................. 132
  7.3 Adding Scantegrity to postal ballots .................... 133
  7.4 Summary ....................................................... 135
  7.5 Discussion ..................................................... 135
  7.6 Future directions ............................................. 136

8 Conclusions and future directions 137
  8.1 Contribution ................................................... 138
  8.2 Future directions ............................................. 139
Foreword

Democracy depends on elections — the people elects those to lead them and to make decisions for them. Any election is the difficult marriage of secrecy and verifiability in that we want all the votes to be secret so that no voter feels intimidated but free to vote according to her own heart and we want the election to be verifiable so that we can all rest assured that the outcome of the election does reflect the will of the people.

Elections depend on people, procedures, software and hardware — people run for office, vote and count the votes and if, in the heat of the moment, they get a chance some of them would cheat to get ahead. To make cheating hard we have put in place procedures that have to be followed: the ballot box is shown to be empty at the start of election day and then it is sealed; ballots are cast into it one by one; at the close of the election the box is signed; it is safely transported to a counting place and only after checking signatures and lists is it opened and finally the votes are counted under close watch from election observers.

Recently voters around the world have been asked to trust that a computer does all these things without any real evidence of them actually doing so.

Elections may fail and we may not pick it up. It may be possible for individuals or groups to cheat in an election, giving the victory to the wrong candidate, but regardless of the number of observers we have it is impossible to be sure. We want independence from people, procedures, software and hardware. We want to be verifiable by voters, media, political parties and anyone else. What we really want is mathematical proof that the election outcome does reflect the will of the people. Because maths you cannot cheat.

If you are thinking “we don’t need that here!” then what about Afghanistan, Iran or the USA where elections have been disputed recently?

We’ll give the voter the power to check that her vote counts. We’ll give anyone the power to check that the final tally is correct. While keeping the vote secret. There is no-one you have to trust, including the current government and system providers. No-one can know your vote including hackers, system operators and poll station staff. Cryptography is what makes this possible.

*From the author’s TEDxStockholm talk in September 2009.*
1 Introduction to Electronic Voting

An electronic voting system is the fragile combination of two opposing requirements: the secrecy of the votes and the verifiability of the election.

In fact, any voting system used in a democratic country aims to strike some balance between these two requirements. Voters must be assured that voting in accordance with their preferences does not pose any risk to them, that it never will pose any risk to them: in short, that it is impossible for anyone to find out how they voted. Voters must also trust that the voting system works correctly and that the result produced is in accordance with the preferences expressed by the voters.

1.1 End-to-end verifiability

Most people are used to voting in a general election by dropping a piece of paper into a ballot box. For a very long time this has been the most transparent, and secret, way to run an election. Transparent because voters (and their representatives) are able to observe the procedures from the casting of the first vote to the counting of the last. Secret because the votes are shuffled in the ballot box, making it impossible for anyone to connect a vote in the ballot box to a particular voter.

Using computers and cryptography it is possible to improve on both these properties. We can make the system completely transparent so that anyone, including voters, political parties, media and external observers, can check that the outcome of the election is correct. The goal is to provide this transparency without violating the secrecy of the votes\(^1\). This may seem counterintuitive but the case is argued in this thesis.

The essence of end-to-end verifiability is that the whole chain from the list of eligible voters, via votes cast and counted through to the outcome of the election is verifiable. Consider again the traditional ballot box and the paper ballot forms cast into it. The average voter comes to the polling station, completes the procedures there, which end in the vote being cast into the ballot box, and leaves. This average voter then has to trust that the box is brought to the place where votes are counted and that all votes in it are, in fact, counted. This kind of trust is needed in the traditional system but we aim to construct electronic voting systems where no such trust is required. Instead the voter is able to verify, after the election, that her vote is among the votes counted and the public is able to verify that the outcome of the election is correctly based on the votes cast — and that those votes were cast by eligible voters. See Section 2.2.3 for a complete definition of end-to-end verifiability.

\(^1\)Secrecy of the votes is not part of the end-to-end verifiability requirement but is a fundamental part of an election.
1.2 Contents

In this thesis we describe the work undertaken as part of the first implementation of Prêt à Voter and draw important lessons from this. Those lessons include our exploration of the problem of implementing Single Transferable Vote without jeopardising the secrecy of the votes, using a number of concurrent polling stations, the problems associated with a human readable paper audit trail, dealing with the democratic right to spoil one’s vote and the problem in paper-based electronic voting systems with the utilisation of the available space on the ballot form.

We introduce a hierarchy of components, organised in four layers, that can be used to model an electronic voting system. The proposal is that the component based model includes information not only from purely technical domains but also non-technical domains such as electoral law in the country where the system is to be used. The proposed model would make it possible, for example, to track changes in a non-technical domain and cascade these into those parts of the system specification that are the basis of the actual implementation of the system. The limitation of the component based approach is that some aspects of electronic voting systems, that appear to be different components, are in fact impossible to separate.

In some countries the number of offices contested in each election and the number of candidates in each of those contests may be quite high and election authorities may therefore look to electronic voting not for the promise of end-to-end verifiability but simply as a mean of helping voters cast their ballots. We therefore introduce a variant of Prêt à Voter that allows voters to create and cast an encrypted vote using a touchscreen interface whilst aiming to preserve verifiability and secrecy assurances of the original system. This makes it possible for the voter to sort the list of candidates as well as searching for candidates instead of reading through a very long list. Furthermore, the proposed scheme reduces the amount of data that must be printed, solving the problem of scanning immensely large ballot forms in the original scheme.

The use of non-verifiable electronic voting systems and their various failures over the last decade has decreased public confidence in the use of any electronic system in elections. The voter verifiable paper audit trail was suggested as a way of auditing the electronic election and it is possible to add such a mechanism to end-to-end verifiable electronic voting systems, but this introduces a number of vulnerabilities. In this thesis we propose a configuration of a human readable paper audit trail that not only allows for the manual recount of any part of the votes cast in the election, but also extends Prêt à Voter with the ability to audit the correct construction of ballot forms that have been used to cast a vote without threatening the secrecy of those votes.

Allowing voters to cast their vote from their home instead of visiting a polling station is seen, in various places, as a way of widening participation. We propose a setup in which various paper-based end-to-end verifiable electronic voting systems can be used to cast postal ballots whilst limiting, as far as it is possible when votes are cast outside of a supervised environment, the threat of vote buying and intimidation of voters.
1.3 Disposition

The proposals made in later chapters of this thesis are based on a review of the state of the art, which is detailed in Chapter 2. The main themes of this thesis are verifiability and usability. In Chapter 3 the work with the first implementation of the end-to-end verifiable electronic voting system Prêt à Voter is detailed, with a series of lessons learned. Those lessons have inspired the work presented in later chapters of this thesis. In Chapter 4 we introduce a component based model for electronic voting systems that allows developers to reduce the security and verifiability analysis of different parts of the system into such analysis of various components. In Chapter 5 we present what can most easily be described as a “front-end” to the Prêt à Voter voting system that aims to be usable and inclusive. In Chapter 6 we build on previous work in human-readable verification of Prêt à Voter and present a paper audit trail which satisfies calls for such auditing capabilities in electronic voting systems but also enables new auditing possibilities not previously envisaged. Finally Chapter 7 describes a set of procedures that aims to limit the secrecy vulnerabilities of postal voting.
2 State of the art

2.1 Introduction

Although electronic/mechanical voting systems have been in use for quite some time, the electronic voting field has really become formalised and thus a more mature area of research in the early 21st century — a time that has seen great controversy in the democratic process around the world.

In this chapter the current state of the art is presented. The effort here is not necessarily to state all the (mathematical) detail but to introduce various results of previous research as tools. The chapter carries references to sources where full details can be found.

2.1.1 Security properties

In the first instance someone who is not particularly familiar with the electronic voting research field may assume that governments and organisations around the world wish either just to save money or, more viciously, to be able to manipulate the result of an election when they consider moving from traditional paper-based systems to electronic. In this review of the current state of the art we will see that although some systems have been devised mainly to fulfill that first requirement and actually do not go very far to protect against the second, the academic research in the field has come a long way to fulfill a set of security properties that in fact cannot be fulfilled by traditional paper-based voting.

The security properties that we wish to achieve in constructing an electronic voting system are described in more detail in Section 2.2.3 but we give the overall goal here as end-to-end verifiability.

2.1.2 Using cryptography to achieve end-to-end verifiability

Although it has been shown by Rivest, in the form of Threeballot [73], that it is possible to construct verifiable voting systems that do not use cryptography, a majority of systems do and those are the systems that we are concerned with here. The goal of these systems that employ cryptography is to achieve end-to-end verifiability by distributing trust among a number of different parties. One such system is Prêt à Voter, which in its first guise [76, 20] relies on RSA [74] and in a later form [85] on Elgamal [26] or Paillier [65].

Most electronic voting systems rely on the calculated risk (or probability) that some participants (i.e. government, clerks, voters, trusted parties, coercers, ...) try to cheat.
Chapter 2. State of the art

Accepting this as a necessary evil, we try to minimise the risk of such threats in the construction of the system. For example, in Prêt à Voter the secrecy of the election is based on a number of trusted parties performing a shuffle. If at least one of the parties performs the shuffle honestly then the secrecy of the election is safeguarded [84, 21] — the conclusion of which is that the addition of more trusted parties will decrease the likelihood of all being dishonest. This distributes the trust in the system, using cryptography, over a number of different parties.

2.2 Elections

Although the intention of this chapter is to detail the state of the art of electronic voting systems it is important to remember that such a system only exists to support the election process, which underpins democracy.

Elections and voting have a long history, as long as that of the notion of democracy in fact, but here we will only look briefly at the history of the secret ballot, some methods used to run elections, the desired properties of any election system and the stages of such systems.

2.2.1 History of secret ballots

Perhaps it suffices in this context to note that progression from no vote, to some people voting, to more people voting, to all people voting is quite a natural progression for an evolving democratic society. Similarly a progression from you giving your vote to someone to cast on your behalf (i.e. someone owning the land you farm or similar) via you being asked for your vote in public to, finally, the secret ballot seems equally natural to those having grown up in a democracy. Each of the steps in this seemingly natural progression has of course been a struggle for those involved.

Until the mid-19th century those eligible to vote would write their preferred candidates for each office on any piece of paper and drop this into the ballot box\(^1\). Although the act of filling out the ballot was done in a voting booth, political parties soon realised that they could influence voters’ choices by printing their own ballots (“tickets” in the USA) or publishing them as advertisements in newspapers [45]. The easiest way for a party loyal voter to vote would thus be to bring this ballot form and cast a vote for the party’s preferred candidates for all offices. Furthermore, political parties soon also realised that if the ballot was printed in such a way that it would be hard to strike out a pre-printed name and write another in its place this would discourage the voter (who was very much entitled to do this) to deviate from the party line, to “split” her ballot. If the ballot was printed on an easily recognised type of paper then party observers at polling stations would also easily be able to check that voters used the “correct” ballot form [45].

\(^1\)This may be true for a host of countries but the United Kingdom and the United States of America are the setting for the history reported in this chapter [45].
To address all these issues the use of a uniform ballot form provided by the election authority was first written into law in the Australian Colony of Tasmania in 1856 and it is thus called an Australian ballot [45, 103]. The purpose of this innovation was to safeguard the secrecy of the vote, a privilege most agree is not only a fundamental right but one without which no democracy can exist.

2.2.2 Methods

Growing up in a democracy one might take the liberties associated with this form of government for granted — and accept the method by which elections are run as quite natural. However, the debate about what method provides the best view of the will of the electorate is ongoing [89]. Although a few methods of running, and most importantly counting, an election are presented here, it is well worth noting that the number of methods is only limited by human imagination. Each government and organisation is free to choose its method and it is important to realise that no single method may be optimal for all societies and organisations. Instead it seems fit to conclude that the democratic process must be applied to forming the electoral system to the electorate and the political system in question.

Although the way votes are counted is arguably a very important aspect of the way the electoral system safeguards the democratic process, there are many other issues that must be considered. For example, the plurality voting system is used to elect one representative of a particular subarea or group called constituency. Although the wishes and beliefs of the members of that group (constituents) might be a complex mix, a single representative is elected. It is therefore likely that if the borders of a constituency are changed (that is to say a different set of constituents belong to the constituency) then this may be reflected in the election of a different representative. Gerrymandering [99] is the term for the action of changing constituency boundaries in order to achieve an advantage in a subsequent election.

Plurality voting and first past the post (FPTP)

Plurality voting is widely used around the world and its purpose is quite straightforward: to elect one representative for each constituency [100]. This election of a single representative is underpinned by some method to interpret the votes cast by voters. The “most common system, used in Canada, India, the UK, and the USA” [100] is what is called first past the post or winner takes all.

As both terms suggest, the winner is the candidate that receives the largest number of votes after each voter has been allowed to cast a vote for exactly one candidate. As can be shown in examples [105, 29, 101, 96] it is quite possible to elect a candidate who an arguably overwhelming majority of voters do not wish to elect. To mitigate this risk it is possible to have a number of rounds. One example of this is the 2007 French presidential election where some 37 million votes were distributed among 12 candidates in the first round and some 35 million given to Nicolas Sarkozy and Ségolène Royal in the second, the previous being elected president with a 53% majority [98].
However, although this may seem to be a major flaw of the electoral system, first past the post has been used to determine the will of the electorate in elections for a very long time. This may well be because it is easy to understand and count. There are ongoing campaigns [89] to change the first past the post scheme used in the UK to some other scheme such as Single Transferable Vote (STV) [89], recently trialled in Scotland [35].

**Single Transferable Vote (STV)**

The foremost virtue of Single Transferable Vote is that fewer votes are wasted. In the plurality vote system where the single candidate with the most votes in a constituency is elected the STV system allows for the election of more than one representative of each constituency [89, 104].

In the plurality vote system a vote for any other candidate than the winner is wasted: it says nothing more of the will of the voter. To address this STV allows the voter to rank one or more candidates in the order of preference. The vote is first awarded to the first preference until that candidate is either eliminated or has achieved the necessary quota to be elected, after which it is awarded to the second preference and so forth.

There are not only a number of options to consider (such as the number of candidates each voter is allowed to rank, what happens if the voter marks more than that allowed number or if she skips a number in the order) but there is also a plethora of STV tallying algorithms to choose from. In its perhaps simplest form the STV algorithm is as follows:

1. At the close of the election all ballot forms are sorted into piles according to the indicated first preference.

2. If any candidate has achieved the necessary quota she is elected. The quota depends on the number of candidates that are to be elected. If the necessary number of candidates have not been elected the votes for this candidate are redistributed onto the other piles according to the next preference and the algorithm reiterates. Variations of STV exist where these votes are weighted so as to, in effect, redistribute only the “surplus” votes not needed by this candidate to win the seat. If the necessary number of candidates are elected the algorithm stops.

3. If no candidate achieves the quota in this round then the candidate with the least number of votes is eliminated and the votes redistributed onto the other piles, after which the algorithm reiterates.

As the STV algorithm is fairly widely used in elections around the world, there exist a number of different versions. For example, when the votes for an elected candidate are redistributed this can be done in a weighted fashion, such as is done in the Republic of Ireland (for Senate elections) and Australia [104, 97]. This method ensures that the electorate does not “waste” votes by giving a candidate more votes than he or she needs to be elected. It does, however, increase the complexity of tallying the election as fractions of votes have to be tracked.
2.2. Elections

Alternative vote (AV) or instant-runoff voting (IRV)

Please consider first past the post (FPTP) and single transferable vote (STV) elections. The first aims to elect a single winner in a multi candidate race and does not consider whether or not the winning candidate achieves a majority. The latter gives each voter the opportunity to transfer her vote in the event that the vote would otherwise be wasted, for example if it is for a candidate who has no chance of winning (and is thus eliminated). STV can also be used to elect any number of candidates in the same election.

In some parts of the world two consecutive but independent rounds of first past the post voting are held, for example the French presidential elections used as an example earlier. This is to ensure that the candidate that wins the election has a majority of the votes: if one candidate achieves a majority in the first round he or she is elected but if not, then a second round with only the two candidates with the most votes in the first round is held. Most likely a set period of time passes between the two rounds during which voters are able to familiarise themselves with the two remaining candidates and are able to decide which one of them to vote for.

However, each round is very much like running separate elections with the complexities and costs related to such an event. Polling stations must be opened, workers employed, ballots transported, aggregated and counted. A solution to this might be to allow each voter to submit both votes at the same time, but as it is not possible to stipulate which two candidates will remain in the final round this may be hard — consider ranked pairs voting [102], which has some similarities to this proposed method. The alternative vote (AV), also called instant-runoff voting (IRV) [95, 89] is a voting method used to solve exactly this.

In AV each voter is able to cast a single ranked vote, how many (all, any, or a maximum specified number) of the candidates a voter is allowed to rank is specified by the rules of the election. In the first round of the election all votes are first given to the first choice on each form. If one candidate achieves a majority then he or she is elected, otherwise the candidate with the least number of votes is eliminated. All those votes are then redistributed to the next candidate in the order of preference. Each redistributed vote is weighted as a full vote, as opposed to in STV where the the redistributed votes may be weighted. This process of elimination and redistribution continues until one candidate achieves the necessary majority to be elected.

One interesting variation of IRV that truly is instant-runoff is the “batch-style” IRV [95]. In this method of tallying the votes are first distributed to the first choice candidates. If no single candidate achieves a majority after this distribution then all candidates but the two with the highest number of votes are eliminated and redistributed to the two remaining, according to who appears closest to the top of the ranking on each ballot.

Range voting and approval voting

In order to solve perceived problems with current election systems, organisations and individual researchers devise new ways to aggregate the will of the electorate that are
then (hopefully) shown to have a set of desirable properties. One such scheme that is not yet in widespread use\(^2\) is Range Voting [29, 101]. In this scheme the voter is asked to give each candidate a numerical score in a pre-agreed range or indicate that she has no opinion of a particular candidate. An example of a range vote ballot form is shown in Table 2.1.

When all voters have expressed their views in this way the candidates are ranked using the average score awarded to them. Furthermore, any candidate who does not achieve at least the score sum of 50% of the highest score sum achieved by any candidate is said not to have reached *quorum* and is eliminated. The election winner is the candidate at the top of this ranked list and if a number of seats are filled by the election the required number of winners are simply taken off the top.

If the range of scores rewardable to candidates by the voters is constrained to “accept or not” then the scheme is in fact called *approval voting* [96]. In this scheme voters are able to vote accept (1) or not (0) for each candidate in a race. After the close of the election the votes received by candidates are counted and the candidate or the candidates with the highest number of votes is/are elected.

### 2.2.3 Desired properties

The desirable properties presented in this section are purposefully listed here and not under a heading specifically concerned with electronic voting systems. Although these may have appeared in the literature specifically as desired properties of electronic voting systems, they are arguably universal.

If the reader would consider any current paper-based system in use today in terms of these properties one interesting aspect may become clear: electronic voting systems do not merely try to emulate adherence of a paper-based system to these desired properties, they wish to add further properties. In the first instance please think about the two verifiability properties: individual verifiability and public verifiability. In a normal paper-based voting system there is little individual verifiability except the possibility to watch the ballot form chain of custody and watch the people and processes in which trust is vested.

If the definitions here deviate from the generally accepted definitions of these properties then a discussion of this is included. All properties (with the exception of eligibility

\(^2\)At the time of writing it seems it is not in use anywhere.
verifiability, which is introduced here) have been defined in [87].

**Individual verifiability**

**Definition.** Each voter is able to verify that her vote is included unaltered in the tally.

The term *verifiability* has previously been defined in two seemingly opposing ways: as “no one can falsify the result of the voting” [30] and as “anyone can verify the correctness of the result” [64] and here we split this definition into two (individual verifiability and public verifiability) and clarify that *verifiability* is only concerned with preventing the falsification of election results by *detecting* errors and malfeasance.

In traditional paper-based voting systems the voter’s ability to check that her vote is included in the final tally is substantially limited. This would involve the voter staying in the polling station after having deposited the vote in the ballot box and subsequently following the ballot box to the counting place and watching all votes in that box being counted. The individual verifiability of the voter is thus only a trust that she has to place in people and procedures and fellow voters helping her to check these.

Individual verifiability is concerned only with the ability of the individual voter to check that her vote is included unaltered in the tally. However, if the voter detects that her vote is not included in the tally or that it has been changed, it is desirable to give her the ability to challenge the election. That would mean that the voter would be able to provide some proof, which can be publicly scrutinised, that her vote has been dropped or changed and thus not counted correctly. Traditional paper-based voting schemes do not do this but we can turn to electronic voting schemes to provide this power to challenge an incorrect election outcome.

This leads to the necessity to provide the voter with a receipt, which may conflict with the receipt-freeness property also described in this section. Such a receipt would have to contain some kind of commitment, which should subsequently be challenged in the event of malfeasance. The discovery that it is in fact possible to provide such a receipt forms the foundation of the electronic voting research field that we know today [9, 10, 17, 81, 76, 20].

As the implementation of the individual verifiability and the receipt varies from system to system (see Section 2.4 for more information on current systems) the production of a receipt can be summarised here in more general terms. Normally we think of a receipt as a piece of paper stating the date of a transaction, who were involved and the contents of the transaction, printed by one of the involved parties and given to the other. In retail the purchase is concluded when the shopper receives the goods in exchange for money and a receipt is printed to document this transaction — if necessary the customer is able to prove to the retailer that an item was purchased from its outlet in a particular transaction.

If this method of producing a receipt was to be implemented in a voting system the voter would ask the election authority (i.e. the voting machine) to register a vote for a

---

\(^3\) ThreeBallot [73] has shown that it is possible to construct non-electronic verifiable voting systems.
particular candidate. The authority would do that, noting down who voted for which candidate in which transaction. Subsequently a receipt would be printed and stamped bearing the message that this particular voter has voted for this particular candidate in this particular transaction. If the voter is later unhappy with the result she may challenge the authority to show where in the tally her vote is. If the authority cannot show that the vote is included in the tally the receipt is sufficient to successfully challenge the election. Clearly this violates the secrecy, receipt-freeness and coercion resistance properties described below.

The breakthrough for receipts in electronic voting came with the encrypted receipt [17], a receipt which does not reveal the contents of the vote until it has been decrypted. The key is that it is possible to decrypt the receipt in such a way that the link between an encrypted receipt and the plaintext vote is broken (see Section 2.3.6). A further major development is the ability to look at the receipt not as a subsequent record of a plaintext transaction but as the vote, the ballot form, itself. Thus, in several electronic voting schemes, the voter produces an encrypted receipt without the involvement of any technology or person (these may violate the secrecy requirement) and this receipt is then recorded and a copy held by the election authority. Thus the receipt is in fact a record of its own transaction.

**Public verifiability**

**Definition.** *It is possible to publicly verify that cast votes have been correctly tallied.*

To explain public verifiability let us first make a distinction between individual and public verifiability. In individual verifiability the voter is able to check the inclusion of her vote in the system input. This is something that it is necessary that each voter does, not only because he or she will be the person with the highest stake in the inclusion of that particular vote but because she may be the only person able to perform this check.

If it may be true that the individual voter is the most highly motivated person to check the inclusion of her particular vote then it may also be true that the electorate as a community is very motivated to verify other aspects of the system. In the case of a traditional paper-based voting system the individual voter would ensure that her ballot goes safely into the ballot box but the electorate as a community would then be motivated to ensure that all the ballot forms in that ballot box are properly counted.

In the context of electronic voting systems the public verifiability would also be to ensure that all votes cast are counted properly, but the manifestation of this will of course differ. In a system that has an input of encrypted receipts (such as will be seen

---

4 The encrypted receipt can take different forms and the form described here is that of Prêt à Voter and Punchscan. An example of a different form of receipt is that of Punchscan II, where the receipt is not an encryption of the vote itself but a value, the knowledge of which gives the voter the opportunity to challenge an incorrectly recorded vote.

5 This assurance would be put forward by any number of publicly appointed experts who are able to perform whatever function, programming or mathematics that is required to verify these parts of the system in question.
in Section 2.4) it is in the interest of the electorate as a community that all encrypted receipts are properly decrypted into plaintext votes. One technique for doing this is using mix networks with random partial checks (Section 2.3.6).

**Eligibility verifiability**

**Definition.** *It can be verified that cast votes have been cast by eligible voters.*

We here introduce the term *eligibility verifiability*, which is a property of a voting system that concerns whether or not it is possible to verify, after the close of the election, that the votes that become the basis for the election result have been cast by eligible voters.

For the traditional paper-based voting system, it is sometimes the case that the local election authority makes available a list of all eligible voters, with marks against those voters who actually cast votes in the election. The voters with a mark against them can be counted and this number compared to the number of votes in the ballot box. That these numbers correspond is evidence, together with the observation of proper procedures in the poll station, that the votes have originated with eligible voters. Making the list of eligible voters, together with a record of whom among them actually cast a vote, publicly scrutinisable makes it possible to reveal mistakes and malfeasance in the form of voters ticked as having voted without actually having done so (which is a common tactic called stuffing of the ballot box) and the exclusion of eligible voters from the list.

In some electronic voting systems such as Prêt à Voter (Section 2.4.2), the voter creates an encrypted vote, which becomes the input to an anonymising mix and decryption. As these encrypted votes do not reveal how the voter has voted they can potentially be noted down next to the voter’s name in the list of eligible voters. This, together with the verification of the anonymising mix, would provide proof that the cast votes were cast by eligible voters, making the system eligibility-verifiable.

**End-to-end verifiability**

**Definition.** *A voting system is end-to-end verifiable if it is individually verifiable, publicly verifiable and eligibility-verifiable.*

In order for an election system to be end-to-end verifiable there must exist no part of it, on which the election outcome relies, that is not verifiable, either by the public or by individual voters. Some definitions of end-to-end verifiable, such as [87], do not include the public verification of the eligibility of the voters who have cast the votes but say that the system has to be verifiable from the point where (encrypted) votes have been cast and published. We argue that without some possibility of verifying the eligibility of the voters who have cast votes, the individual verifiability of the inclusion of individual

---

6This, however, enables the forced abstention attack, see Section 2.2.3 on coercion resistance and Section 2.4.7 on “everlasting privacy”. 

---

15
votes (see below), together with public verifiability of decryption and tallying, does not constitute a truly end-to-end verifiable voting system, a view supported by [49].

The “ends” between which the system is verifiable is thus (a list of) eligible voters on one side and election outcome on the other, with a verifiable chain between the two.

End-to-end verifiability is related to software independence, introduced by Rivest and Wack [75]. A voting system is software independent if “an undetected change or error in its software cannot cause an undetectable change or error in an election outcome.” An end-to-end verifiable system must be software independent but in some instances a software independent system is not end-to-end verifiable as defined here (Rivest and Wack use a DRE with a voter verifiable paper trail as example: the malfunction of such a system would be detected by auditing procedures but the correct recording and counting of each vote is not proven as “default”). An end-to-end verifiable system must also be verifiable in other, non-software, parts of the system so that, for example, undetected change or error in procedures carried out by humans cannot cause an undetectable change or error in the election outcome.

**Fairness**

**Definition (Fairness of opportunity).** *All eligible voters are able to cast their votes.*

**Definition (Fairness of information).** *All voters have access to the same relevant information.*

Fairness has previously defined as “nothing must affect the voting” [30] and “no one knows any intermediate result before the deadline has passed” [64] and here we wish to bring together both these notions into ensuring that all eligible voters are able to cast their votes with access to the same relevant information. This requires that no partial results are made available before the close of the election as revealing such information would not only influence remaining voters but also give them an unfair advantage against other voters as they are able to strategically change their vote.

Although the fairness of the election is outside the scope of most current verifiable electronic voting schemes, there are avenues for technical remedies to some of the associated issues. For example, in order to make sure that each voter is given exactly one vote the encrypted vote cast by each voter (see Section 2.4) can be published alongside the name of the voter on the web bulletin board, as the encrypted vote does not reveal the contents of the vote. All encrypted votes that are subsequently decrypted into plaintext votes must thus be associated with a voter and that voter is able to ensure that that receipt is hers and no other.

---

7See previous footnote about the forced abstention attack.
8This does not introduce a measure for a voter who has not voted but found that a receipt is placed against her name to prove that she has not voted and the receipt has been erroneously placed next to her name. It also enables the forced abstention attack whereby a coercer forces a voter not to vote.
Integrity

Definition. All votes are cast as intended, recorded as cast and counted as recorded.

In essence, integrity is the property that implies that the announced result of the election reflects the will of the people. This definition builds on reasoning around this property by [11] and [12]. It is common to consider whether the vote has been cast as intended, then recorded as cast and counted as recorded. This creates a chain from the voter and her intent to the result of the election. Let us look at each of these properties in a little more depth.

Cast as intended What is the probability that the voter has been able to correctly express her intent on the ballot form? The cast as intended property encapsulates the disparity between what the voter is intending to vote for (her idea of which candidate she wishes should win the election) and the vote she is able to produce.

It may seem that electronic voting systems are particularly likely to suffer from problems in this area — but problems of this nature do invariably invade all voting systems, be they electronic or not. Consider the traditional paper ballot for example. In an election that uses the first past the post principle to select its winner the voter may be asked to enter, in a table containing all candidates in the race, a cross next to the candidate she wishes to cast a vote for. This may seem trivial to most but those unable to read the instructions or who choose not to do so may opt to make a tick mark next to the preferred candidate. This might not be permissible by local electoral legislation and the voter has thus not been able to vote as intended.

Next consider an election that does use a paper ballot but the single transferable vote (STV) method for selecting the winner. In many cases the voter may wish to vote only for a single candidate and unless this has been made quite clear in written or oral instructions, she might make a cross or tick mark instead of the digit “1” that legislation demands.

From these examples it is becoming clear that all voting systems must consider usability aspects to ensure that voters are able to cast their votes as intended. As in many issues concerning such large-scale systems as a voting system it is invariably so that the cast as intended property is measured in some proportion of all voters who are able to correctly express their will in the election. One may note here that it therefore should be possible to benchmark voting systems on this property.

Recorded as cast When the voter has expressed her will by filling out the ballot form (or some other action needed for her to do so) the next step towards the accumulation of the will of the electorate is to ensure that all votes are entered into the system correctly.

In the case of the traditional paper-based system this of course means that the ballot form enters the system by being dropped into the ballot box.

In the case of an electronic voting system (where the “dropping” of the ballot form into the box is actually the recording of it) this becomes slightly more complicated.
Let us consider a direct recording device (DRE), which in short can be described as a computer that the voter interacts with in order to cast her vote. The computer may for example display all candidates in a particular race on a touch sensitive screen and the voter casts a vote by simply pressing one of these. The usability related issues of the DRE, i.e. any problems the voter may have to use the machine to express her intention, belong to the cast as intended property. However, when the voter has expressed her vote (accurately or not) then whether the vote is stored and aggregated as a vote with precisely the expressed intention, or not, is a recorded as cast issue.

A DRE is a particularly good example here because they do not normally give the voter (or the election authority) any proof that the vote has been correctly recorded. If, for example, the voter presses the name of a candidate $G$ and the DRE displays a message to thank the voter for casting her vote there is no guarantee that the machine has not malfunctioned (or been maliciously programmed) and recorded a vote for a different candidate $B$.

**Counted as recorded** One may argue that all propagation, transportation and aggregation issues are part of the counted as recorded property because the recording of the vote is done exactly once and subsequent steps in the election procedure may be viewed as part of counting. In the traditional paper-based voting system this involves transporting the ballot boxes in a certain region to some counting facility, opening them and then counting the votes. The election authority is concerned with making sure that all ballot boxes reach the counting facility, that all votes are properly counted and that the results are accurately reported to some central point and in a timely fashion.

In the case of an electronic voting system each of these steps may be done electronically, perhaps even over a network. In the chain from the voter via the recording of the vote in the computer, the DRE must allow its collected votes to be aggregated, perhaps over a network or using some mobile storage media. As previously mentioned the DRE does not give any proof that its counters truly reflect the votes cast and it is now worth noting that there are similarly no proof or guarantees of the proper function of the transportation or aggregation of these counters.

**Secrecy (aka Privacy)**

**Definition.** Given a particular vote, it is not possible to determine by which voter it has been cast.

As seen previously (Section 2.2.1) the secret vote is the result of a lengthy electoral development and a very important property of election systems that is widely understood. In simple terms it must be impossible for anyone (including everyone from spouses via poll station workers and party officials to the election authority and the government) to find out how a particular voter has voted\(^9\). Secrecy has been defined as “all votes must be secret” [30] and later “it is infeasible to associate individual votes and voters” [64]

\(^9\)Note the problems of guaranteeing that a vote is secret for ever — see also Section 2.4.7
but a more formal definition is given by [50]: given two voters $A_1$ and $A_2$ and two votes $v_1$ and $v_2$ it is not possible to distinguish between the two cases

1. $A_1$ casts $v_1$ and $A_2$ casts $v_2$
2. $A_1$ casts $v_2$ and $A_2$ casts $v_1$

In the case where a coercer can find out how you have voted, i.e. when the secrecy of the vote is compromised, it is easy to imagine cases where you may be influenced to vote in a particular way. You may become unable to vote for your preferred candidate because of fear of violence or of the influence of a monetary or other award. Only in the case where the vote is unconditionally secret can the outcome of an election be regarded as a fair and a true representation of the will of the electorate.

In the traditional paper-based voting system the ballot box is used to safeguard the privacy of the voter. All voters place their ballot forms in the same ballot box (that may then be shaken, ensuring that no-one is able to tell which ballot form belongs to which voter). The introduction of the uniform, pre-printed Australian ballot (see Section 2.2.1) was very much with the same intention and made it possible to legislate that the voter must not make any distinguishing mark on her form such that someone can decisively say that a ballot form drawn from the ballot box belongs to that particular voter.

The integrity and fairness of any voting system is quite a different matter than the secrecy of the vote. The accuracy and the secrecy are often regarded as properties that are provided at the same time but they are in fact so different that they not only have different solutions but that the unconditional provision of one may severely limit the extent at which the other can be provided, see Section 2.4.7.

As an interesting note at this stage it is possible to look at Prêt à Voter (Section 2.4.2), which uses a number of trusted parties to ensure the secrecy of the vote — each trusted party breaks the link from vote to voter. The calculations of the probability by which the secrecy of the vote is safeguarded are based on a cryptographic scheme being unbreakable, such as RSA (Section 2.3.1) or Elgamal (Section 2.3.2), which they may well be at the time of the election. In the same spirit one may look at the hashes used by Punchscan (Section 2.4.3) that at the time of the election must be regarded as “unbreakable”. However, if one looks at the development of computing in the past it seems likely that at some point after the election will the cryptography used become breakable. At that moment the secrecy of the vote completely ceases and even though this may be ten, 20 or 30 years after the election was held, the threat of this “end” to the secrecy of the election may be enough to influence the voters in one way or the other. This has led to the introduction of the concept of everlasting secrecy (see Section 2.4.7).

**Receipt-freeness**

**Definition.** A voter is not able to prove to a third party how she has voted.

As we have seen in the previous section detailing the individual verifiability property it is necessary to give each voter a receipt to allow them to check that their respective
votes are included in the final tally. However, the receipt-freeness [10] property is so named because it requires that the voter is unable to prove to anyone the contents of the vote, using the receipt.

To explain why this is necessary we should start by exploring the consequences of giving a voter a receipt that states how she has voted. In a scheme where such a receipt is issued the possibilities of coercing a voter are abundant:

- The coercer offers the voter a sum of money to present a receipt with a vote for a particular candidate (vote buying).
- The coercer threatens to harm the voter unless she presents a receipt with a vote for a particular candidate.
- An employer requires that employees present receipts of votes for a particular candidate.
- There may be pressure within the family or social group to vote in a particular way; it may be a matter of honour to show others the receipt.

Please note that with the Internet vote buying could flourish at unprecedented levels: a coercer can ask voters to e-mail a digital photograph of the receipt, promising to respond with a PayPal online cheque. The coercer can thus target a very large proportion of the population at a low cost.

**Coercion resistance**

**Definition.** A coercer who interacts with the voter before and during the voting process is unable to distinguish between the voter’s compliance and non-compliance.

Extending the receipt-freeness property, a coercion resistant electronic voting system, as defined by [46], would ensure that a coercer that can interact with the voter during the voting process cannot find out if the voter has voted in the way the coercer demanded. This includes that the coercer cannot force a voter to cast a random vote or to spoil her vote. This is closely linked to receipt-freeness because if a receipt divulges, for example, that the vote has been purposefully spoilt, then a coercer can demand to see such a receipt (this is called a forced abstention attack).

The definition here builds on the long definition provided by [46], part of which reads

A coercion resistant scheme offers not only receipt-freeness, but also defense against randomization, forced-abstention, and simulation attacks [...]
and this definition of the coercion resistance property regards the inability of a coercer to distinguish between the case where the voter does comply with the coercer’s request to vote in a particular way and the case where she does not.

As we have discussed the necessity to issue a receipt above we can now consider only systems that do. However, it is important to note that any leak of information through the receipt should be considered a violation of the coercion resistance property, not only information about which candidate the vote is for, which is a violation of the receipt-freeness property. There are other leaks such as the following:

- The receipt must not reveal if the voter has in fact purposefully spoilt her vote: spoiling one’s vote is, in some jurisdictions, seen as a fundamental right. The coercer would here demand that the voter presents a receipt of a spoilt vote (forced abstention attack). If the coercer aims this attack against members of a different party this would be beneficial to the coercer’s party.

- If the voter opted not to vote in all races contested in the same election this should not be possible to deduce from the receipt. Here the coercer would demand that the voter presents a receipt without a vote in one or more races.

Finally it is worth noting that of course the existence of a receipt indicates that the voter has attended the polling station and cast a vote — even if it may be possible to spoil the vote. A voter should thus be able to destroy the receipt to hide it from a coercer who is forcing her to abstain.

The relationship between Secrecy, Receipt-freeness and Coercion resistance

The properties secrecy, receipt-freeness and coercion resistance are related in an increasing level of strength of attack that a coercer exerts. The secrecy property is concerned with what the coercer can simply observe by watching the voting system and published data from it. The receipt-freeness property is also concerned with data that the coercer can receive from the voter after she has cast her vote, which may be in the form of a receipt. Finally the coercion resistance property also considers instructions that the coercer can give the voter, and other interactions between the coercer and voter, before and during the voting process.

Robustness

**Definition.** A system is able to withstand and/or can successfully recover in a timely manner from the failure of hardware, software and procedures and detected cheating.

The robustness of a system is its ability to function in the face of various failures, such as hardware or software failure or active attempts to disrupt its function. The verifiability of an electronic voting system is heavily dependent on the robustness of the same system: if the system can be disrupted then it does not matter if the integrity of individual votes can be confirmed. However, taking one step away from electronic
voting and considering elections and voting in general is important here. Any election system must be robust so as to ensure that all eligible voters are able to cast their votes. It then naturally follows that the system must be sufficiently robust to count all those cast votes.

With robustness comes recovery strategies, answering questions such as these:

- What happens if a fire breaks out in a polling station?
- What happens if ballot boxes are broken open by a golf club wielding man?\(^\text{12}\)

Using electronic voting there are of course further issues to contemplate, such as:

- If one of the servers crashes, will the election keep running?
- If one of the parties involved stages a denial of service attack, what effects will this have and are these acceptable?

### 2.2.4 Stages

As any election is an immense, costly project it is quite beneficial to consider its different stages [6]. Each phase requires a set of tests to be executed to ensure that the election does not proceed to the next phase until postconditions of the preceding phase as well as preconditions of the succeeding phase are met.

#### Set-up phase

The set-up phase creates the preconditions for the election. The following are issues addressed during this phase:

- **Nomination of candidates** — the candidates wishing to run for a position are nominated and they campaign to raise awareness of their agenda in the electorate.

- **Election system set up** — regardless of whether the system used to run the election is electronic or not, there are many set-up issues to consider. The definition of a ballot form must be agreed at some early stage to allow these to be printed and distributed to polling stations or voters in the case of the paper-based voting system and computer systems must be loaded with the correct definitions, tested and distributed to polling stations in the case of the electronic system.

- **Rules agreed** — in order to run an election with the desired properties we have seen previously it is necessary to agree a set of rules before the start of the election. To this set belong issues regarding not only such things as the ordering of candidates on the ballot form and the correct way of filling the form out but also much more detailed instructions for poll workers and others on how to behave, what conduct is expected and how to act in a range of situations.

\(^{12}\)Reference to news story from the 2007 Scottish elections where a man attacked staff, furniture and ballot boxes with a golf club in Edinburgh.
2.2. Elections

- **Publication** — it is important before proceeding that all rules, instructions, configurations and the like constructed during this phase be published so as to allow them to be publicly scrutinised.

**Registration phase**

As stipulated by electoral law each eligible voter is either pre-registered or is able to register to vote. After identification and accreditation the lists of accredited voters are aggregated and distributed to the correct constituencies and poll stations.

Registration of voters is normally done by some election authority and there is normally only one such authority, making the process non-transparent and (somewhat) unsafe: consider societies where the current government may be opposed to certain ethnic or religious groups voting in an election.

**Voting phase**

When the process of running an election reaches this phase it is beneficial to run a kind of “self test”: check that the two previous phases have completed successfully and that stake holders (voters, candidates, government, ...) agree the election is set to run in accordance with all the desired properties (see Section 2.2.3) before proceeding to open the election. Of course normally the opening hour of the election is agreed and published in the set-up phase — and one might assume that if the election process reaches this third phase then cost and political pressure will prohibit a delay.

The voting phase normally takes place during a fairly short period of one or perhaps a few days, “election day”. Voters are able to go to their local poll station, identify themselves and cast a vote. The phase includes the collection and anonymisation of votes.

**Tallying phase**

When votes have been cast and the predetermined vote casting period has ended the votes are transported to a central counting facility where they are tallied by employees or volunteers. This must be done in strict accordance with the rules determined in the set-up phase. When this work is complete the required number of winners of each race in the election have been elected.

### 2.2.5 Threats

There may always be people who wish to change the outcome of an election by undemocratic means and in this section we take a brief look at those involved in the running of an electronic voting system and what threat they pose to the correct function of that system.
Chapter 2. State of the art

The coercer

The coercer is someone actively trying to influence a voter to vote in a particular way. In our view the coercer is not adverse to doing anything that is possibly in his power to make a voter vote according to the coercer’s choices: bribery, threat, confusion, misinformation... Whenever the opportunity would arise for the coercer to influence a voter to give her vote to the coercer’s favourite candidate or to take her vote (see randomisation attack below) from a different party that is likely to do better than the coercer’s, he will.

It only makes sense for the coercer to offer a monetary reward to a voter, for voting in a specific way, if the coercer can get some proof that the voter has complied with these instructions. In a secret election there are procedures and technology that is put in place to make sure that the coercer is unable to get at such evidence. This is, for example, the reason why votes are filled out by voters in polling booths and why most countries would consider a vote spoiled if there are any other marks than the ones necessary to express the vote on the ballot form. If such marks were allowed then the coercer would ask the voter to write her name on the ballot form and he would be able to inspect the vote as it appears in public during the counting.

As we have already seen, it is the goal of an end-to-end verifiable electronic voting system to be coercion resistant. This requirement is more than just making it impossible for the coercer to find out how a particular voter has voted, the goal is also to make it impossible for the coercer to force the voter to abstend from voting (the forced abstention attack), to spoil her vote or to give the vote to a random candidate. If, as an example, the system requires a list of all voters who have voted to be published after the election so that it is possible to verify that the votes that have been counted have all originated in eligible voters and have not been added by someone else, then the coercer can give the voter a monetary reward, or threaten her, to ensure that she does not vote and thus does not appear in the list of people who have voted.

The problems associated with the coercer are further explored in Section 2.4.7.

The trusted party

As we see in later parts of this chapter, end-to-end verifiable electronic voting systems aim to distribute the trust in the system over a number of different trusted parties. The secrecy and integrity of the election is therefore not in the hands of a single person or a single organisation but rather in the hands of a number of parties that all have to work together to violate the secrecy or integrity of the election. It is therefore important that these trusted parties are selected in such a way that they are unlikely to work together: they may for example be drawn from the full political spectrum and include both international organisations and established election observers.

The general understanding of a trusted party is thus that they are given the opportunity to safeguard the votes of those who put their trust in them. So for example, if a specific political party becomes one of the many trusted parties in the running of the election, the party would violate the trust that their members place in them to keep
their votes secret and secure if they cheat or leak any secret information. Such cheating or leaking would not be detrimental to the election (if not a large enough number of other trusted parties do the same) but it is in the party’s interest not to let down their members.

To proceed with this example, it may also be in the interest of this specific political party’s members to have the party try to cheat, allowing the own candidate to win. This attempt will still not be detrimental to the election in this setup, but this is now how we regard the trusted parties: we give them the opportunity to do the right thing, but if they decide not to do that then they should be found out, they should not be able to influence the outcome of the election and they should not be able to break the secrecy of the election.

The malicious voter

As discussed above, the coercer would in most cases require some kind of proof that the voter has complied with whatever the coercer wishes her to do. Such proof may not always be straightforwardly available (even though it in some cases might be published as part of the verifiability of the election) so the coercer may require the voter to create it. Consider for example that the voter takes a camera with her into the polling booth. As she is shielded from view no-one can see that she is taking photos of the ballot form, a voting machine that displays a confirmation screen, or anything else that may give the coercer indication that the voter is complying. It is of course the case that the voter may feel she has no choice but to comply if the coercer is threatening her, but if she is merely offered a financial reward in exchange for this then she has a choice but still chooses to break the secrecy of her vote.

Furthermore, the goal of most end-to-end verifiable election systems is to give the voter a possibility not only to check the inclusion of her vote in the final tally but also challenge the election if she feels that her vote has been changed or not included in the tally. If a voter that, for example, favours a small political party wishes to cause problems in the election she could potentially do this by claiming that her vote was changed, when in fact it was not. The claim that the vote was not correctly recorded, even in the face of evidence to the contrary (which end-to-end verifiable systems aim to provide), may be enough to cause widespread distrust in the system and may impede the democratic process.

For these reasons we must consider the voter to be “malicious” and not only make it impossible for her to break the secrecy and integrity of the whole election but also that of her own vote. We must also ensure that the system can refute false claims and support correct claims of malfunction.

2.3 Cryptography

This section aims not at providing a complete reference on the cryptographic systems introduced here but a discussion of the key properties of these systems as a foundation
to later chapters of this thesis.

In electronic voting systems cryptography is not merely used to hide something that is secret and it is not simply extended to provide authentication features — this is standard use of cryptography today. Cryptography in electronic voting is used in a large, complex web of encryptions, decryptions, signatures, checks and proofs. It is used to allow a large, complex, distributed system to prove to various people (voters, media, observers, ...) that it is working as it is intended to work. Perhaps even more importantly it is used to hide from the system itself the data that it works on.

2.3.1 RSA

Invented\textsuperscript{13} in 1977\textsuperscript{14}, the Rivest, Shamir and Adleman (RSA) encryption algorithm \cite{Rivest1978} provides public-key cryptography that can be used both for the encryption of messages and the signing of the same \cite{Blum1978}. “Public-key” implies that the encryption scheme uses key pairs, a \textit{public key} normally used to encrypt messages and a \textit{private (secret) key} normally used to decrypt messages. Public key cryptography thus enables me to set up a key pair and give you the public key quite openly. When you use this key to encrypt a message only I can decrypt it.

The security of RSA is based on the difficulty in factoring very large numbers into their constituent primes\textsuperscript{15}. When two very large primes are chosen and multiplied together it is a hard task, even for a computer, to find those two primes. It would of course be possible to eventually find them by using a brute force method (such as simply going through all the primes in order and testing if they divide the number in question) but it would take a long time. If the primes are chosen to be very large, the security of RSA is based on the belief that it might, for example, take all the computers in the world working together thousands, or even millions, of years to find a particular pair of primes.

2.3.2 Elgamal

The Elgamal \cite{ElGamal1985} cipher system is, like RSA, a public key system, meaning that one has a private key that is secret and a corresponding public key that can be freely distributed. A message encrypted with the public key can only be decrypted with the private key. One difference between RSA and Elgamal is that the previous is \textit{deterministic}: every time a particular message is encrypted with the same key, the resulting ciphertext will look the same. Elgamal on the other hand is \textit{probabilistic} and so two encryptions of the same message under the same key may look different.

\footnotesize
\textsuperscript{13}The same cryptographic scheme was in fact invented by Clifford Cocks and James H. Ellis in 1973 but because the two men worked for the British government this discovery was kept secret until 1997 \cite{Cocks1973}.

\textsuperscript{14}The story is well retold in \cite{Rivest1999}.

\textsuperscript{15}A prime number is a number that can only be divided by 1 and itself. Therefore 2, 3, 5 and 7 are examples of prime numbers: there exists no other numbers that divide them except 1 and the respective number.
The secrecy of Elgamal is based on the difficulty in computing discrete logarithms. As we are using certain aspects of Elgamal cryptography in Chapter 6 we give more detail on this scheme.

Generating a key pair

1. A group $G_q$ is a subgroup of $Z_p^*$ of order $q$. $g$ is a generator of this group.

2. A random value $x$ is chosen from $\{0, \ldots, q - 1\}$

3. The value $h$ is computed as $h = g^x$

4. The public key is $(G, q, g, h)$

5. The private key is $x$

Encryption

Using the recipient’s public key the sender is able to calculate a shared secret $s$ based on the message he wishes to send. This message must be one element of the predefined group $G$. A common strategy for sending messages with any content is to use a symmetric crypto system to encrypt the message and to use a random element from the group $G$ as the key for this encryption.

1. A random value $y$ is chosen from $\{0, \ldots, q - 1\}$

2. $c_1$ is calculated as $c_1 = g^y$

3. The shared secret $s$ is calculated as $s = h^y$

4. The message $m$ must be a element of the group $G$

5. $c_2$ is calculated as $c_2 = m \cdot s$

6. The ciphertext is $(c_1, c_2)$

Decryption

1. The ciphertext $(c_1, c_2)$ is received

2. The shared secret $s$ is calculated as $s = c_1^y$

3. The message $m$ is computed as $m = c_2 \cdot s^{-1}$
Re-encryption

Because of the probabilistic property of the Elgamal cipher it is possible to change the ciphertext without changing the plaintext within. Knowledge of the private key is not necessary to do this, only of the public key.

1. The ciphertext is \((c_1, c_2)\) and the public key \((G, q, g, h)\)
2. A random value \(s\) is selected from \(Z_q\)
3. The re-encrypted ciphertext \((c'_1, c'_2)\) is computed as \((c'_1, c'_2) = (c_1 \cdot y^s, c_2 \cdot g^s)\)

Threshold Elgamal

Given a large prime \(p\) and a generator \(\alpha\) of a \(q\)-order subgroup of \(Z^*_p\). A party \(A\) chooses a secret key \(k\) and computes \(\beta\):

\[
\beta := \alpha^k \pmod p
\]

The public key is \(p, \alpha\) and \(\beta\). \(k\) is the secret key. Encryption of \(m\) yields a pair of terms computed thus:

\[
c := (y_1, y_2) := (\alpha^r, m \cdot \beta^r) \pmod p
\]

where \(r\) is chosen at random. \(A\) decrypts \(c\) as follows:

\[
m = y_2 / y_1^k \pmod p
\]

The security of Elgamal rests on the presumed difficulty of taking discrete logs in a finite field. Thus, recovering the secret \(k\) exponent from knowledge of \(p, \alpha\) and \(\beta\) is thought to be intractable.

A randomising algorithm like Elgamal allows the possibility of re-encryption: anyone who knows the public key can re-randomise the original encryption with a new random value \(r'\):

\[
(y'_1, y'_2) := (\alpha^{r' + r}, \beta^{r' + r} \cdot m)
\]

which gives:

\[
(y'_1, y'_2) := (\alpha^{r' + r}, \beta^{r' + r} \cdot m)
\]

Clearly, this is equivalent to simply encrypting \(m\) with the randomisation \(r + r'\) and decryption is performed exactly as before. We will see the utility of re-encryption when we come to describe anonymising mixes. Note that, crucially, the device performing the re-encryption does not use any secret keys and at no point in the re-encryption process is the plaintext revealed.

In fact we will use exponential Elgamal, where \(m\) is encrypted as:

\[
c := (y_1, y_2) := (\alpha^r, \alpha^m \cdot \beta^r) \pmod p
\]

Thus the plaintext is carried in the exponent of \(\alpha\).
2.3.3 Paillier

The Paillier [65] cipher is a public key cryptography scheme based on the difficulty in computing n-th residue classes. The scheme is additive homomorphic, meaning that knowledge of the public key and the encryptions under that key of two messages, $E(m_1)$ and $E(m_2)$, allows you to compute the encryption of the sum of those messages, $E(m_1 + m_2)$.

2.3.4 Digital signatures

A digital signature scheme is used to prove the origin of a message. Normally the process starts with the calculation of a message digest (commonly hash), which is a value of some predetermined length that is calculated on all the bits of the message such that if a single bit changes then the message digest will change. A hash algorithm is also called a one-way function, meaning that a message $m$ is input into the function and a message digest $m_d$ is output but given a digest $m_d$ it is very hard and/or time consuming to find the corresponding $m$. It is the goal of a hash algorithm to cascade a small change in the message into a large change in the message digest, making it difficult to find two messages $m_1$ and $m_2$ that have the same message digest $m_d$.

As the message digest is commonly much shorter than the message itself, it is less computationally expensive to apply a cryptographic signing algorithm to this digest than to the whole message. The signed message digest is appended to the message when it is sent to its recipient and this is commonly known as a digital signature. The recipient recalculates the message digest from the message and compares the signed message digest to this. If this check holds then it proceeds to check that the signature on the message digest is indeed that of the alleged originator, a process fully dependent on the cryptographic signature scheme.

2.3.5 Threshold cryptography

There are extensions to cryptographic schemes such as Elgamal [67] and Paillier that allows the secret key to be split into several parts. Normally these parts would be held by $n$ different persons or organisations and the scheme would be set up in such a way that a threshold number $k$ of the parties must work together to perform a desired cryptographic function. The benefit from such a setup is that if a message is encrypted using the public key, the plaintext can only be recovered if $k$ key holders work together. This allows a group to enforce an agreement, for example one not to decrypt any ciphertexts before a particular point in time. If fewer than $k$ key holders decide to cheat then they cannot successfully decrypt any ciphertexts but of course, if $k$ or more keyholders decide to cheat they will be successful.

For electronic voting this is very useful in two different ways:

1. Enforce agreement not to decrypt. The secrecy and accuracy of a number of electronic voting schemes described in this thesis is dependent on some ciphertexts not being decrypted until after a particular point in time (the close of the election)
or never being decrypted. In a scenario where a single entity holds the only key
to these ciphertexts that entity must be fully trusted not to break the rules. In a
scenario where this trust is distributed over $n$ parties then the likelihood that $k$ of
these work together to cheat may drop. From the point of view of a voter this may
mean that even if he or she does not trust party A, which holds one key share not
to cheat, it may trust party B, which holds another. The likelihood that parties
A and B will work together is judged very small as they are political enemies.

2. Load balancing/robustness. In the scenario where a single entity is responsible for
the only key there may be events (malicious or non-malicious) that stops it from
performing a proper function: anything from the accidental destruction (loss) of
the private key to a deliberate denial of service attack. If the key is shared among
several political parties for example, the loss of a key share belonging to a political
party that is not happy with the likely outcome of the election does not stop
the system from functioning correctly. It may also be possible to use threshold
cryptography to perform load balancing as the $k$ out of $n$ available key shares does
not have to be the same $k$ each time.

2.3.6 Mix networks

Mix networks were first introduced by David Chaum [16] and can be seen as a way of
passing a message $m$ across a network of trusted parties $T_n$ such that each trusted party
$T$ contributes to the hiding of the identity of the sender. Although first envisaged by
Chaum as a way of sending and responding to e-mail anonymously across the Internet,
he also suggested that it is applicable in electronic voting protocols.

Decryption mix networks

The first devised mix network was a decryption mix network [16] where the message is
hidden under several layers of encryption and where each layer $L_x$ is encrypted under
the public key $PK_{T_x}$ of a particular node $T_x$. As the message passes to a node in the
network it removes its layer of encryption, effectively altering the look of the message,
before passing the resulting encrypted message on to the next node $T_{x+1}$.

In a decryption mix network it is thus only possible to regain the plaintext message if
the encrypted message passes through all nodes in the order specified by the layers. The
anonymisation strategy is based on the likelihood that at least one node of those through
which the message has passed will not leak from where it was passed the message. In
other words, if the eventual recipient of the message attempts to trace the message back
through the nodes of the network he will fail as soon as a node does not leak such
information. This is in fact the anonymisation strategy used by Tor [91, 72].

Chaum proceeded to use the decryption mix network in a scheme incorporating visual
cryptography [17] comparing the multi-layered encrypted values to russian dolls, which
consist of a series of different-size dolls, each contained within the next larger size doll.
Although a simplified version of the scheme was introduced by Ryan [15, 81, 76] and
named Prêt à Voter, the multi-layered encrypted values were still called Russian dolls. This was changed to another suitable metaphor, namely the onion, which has many layers when Chaum, Ryan and Schneider [20] published a joint paper on the simplified scheme now definitely named Prêt à Voter. Here the trusted parties (nodes in the mix network) are also named tellers.

In this new scheme [20] each layer of the onion holds a germ $g$ and each teller is responsible for two layers of this onion, i.e. the teller holds the private key of two consecutive encrypted layers of the onion. With $D_0$ a “nonce-like value” the construction of the onion is thus:

\begin{equation}
\{g_{2k-1}; \{g_{2k-2}; \{\ldots, \{g_1; \{g_0, D_0\}_PK_{T_0}\}_PK_{T_1}\}_PK_{T_{2k-2}}\}_PK_{T_{2k-1}}\}
\end{equation}

(2.1)

The secret $\theta$, i.e. the cyclic shift of the candidate list (see Section 2.4.2), is in fact the sum of the hashes of each germ value $g$ taken modulo the length of the cycle $v$:

\begin{equation}
d_i := \text{hash}(g_i) \pmod{v} \quad i = 0, 1, 2, \ldots, 2k - 1
\end{equation}

(2.2)

\begin{equation}
\theta := \sum_{i=0}^{2k-1} d_i \pmod{v}
\end{equation}

(2.3)

In order to explain the decryption mix network, let us think of an encrypted receipt as the combination of an onion with an indication of the position of the voter’s mark. The purpose of a mix network in electronic voting is to hide\(^\text{16}\) the link between an encrypted receipt, which the voter is allowed to take home, and the decrypted plaintext vote so as to provide a receipt but preserving the receipt-freeness property of the system (see Section 2.2.3).

A Chaumian mix, referring to the mix steps introduced in [17] and developed in [20], for a teller $T$ can be described with the following steps:

1. $T$ receives all encrypted receipts in the batch $B_{\text{input}}$
2. One layer of encryption is removed from each encrypted receipt $R$ and the nonce $n_1$ is regained
3. The position of the voter’s mark is reversed as indicated by $\text{hash}(n_1)$
4. The complete batch of encrypted receipts now form batch $B_{\text{middle}}$, which is published
5. Another layer of encryption is removed from $R$ and its nonce $n_2$ is regained
6. The position of the voter’s mark is again reversed as indicated by $\text{hash}(n_2)$
7. The resulting batch of all encrypted receipts form batch $B_{\text{output}}$, which is published

\(^{16}\)The word “hide” was chosen here to show that there must be a link that can be audited, but that link should be hidden by the tellers.
Further detail of the Prêt à Voter scheme is found in Section 2.4.2 so here it suffices to mention that the result of these consecutive decryptions and reversed shift of the position of the voter’s mark is a final batch $B_{output}$ where the position of the voter’s mark is related to a candidate list in the base order. Thus, the output from the decryption mix network is a set of plaintext, countable votes.

At this stage it is beneficial to note that this decryption mix would be perfect in the case where all tellers are truly reliable. However, in order to find any errors and provide an incentive for the tellers to remain trustworthy the mix network must be audited. One audit method is the random partial check, described below.

**Re-encryption mix networks**

The purpose of a re-encryption mix network [66, 1, 39, 40, 2, 42, 31, 62, 14, 34, 33, 41, 32] is very much the same as in a decryption mix network, to hide the link between the encrypted receipt and the plaintext, countable vote. However, the setup is slightly different. The general actions of each node in the mix network is to shuffle all the encrypted receipts in the batch and then to hide that shuffling by changing not the contents but the look of the receipts. In the decryption mix network this is done by removing a layer of encryption. In the re-encryption mix network the look of a receipt is changed by the injection of randomness into the encryption.

In the mixing phase of the re-encryption mix network the batch of all encrypted receipts is passed between a set of tellers (also called clerks [85]) that each shuffles the receipts and re-encrypts them. As in the decryption mix network where the encrypted receipts are turned into plaintext votes by repeated decryption the encrypted receipts in the re-encryption mix network must also be turned into plaintext votes that can be counted. This is done in the decryption phase where the last output batch of encrypted receipts is decrypted once to regain the plaintext votes.

As the person or organisation that holds the private key needed to decrypt the last batch of encrypted receipts can in fact decrypt any and all encrypted receipts in all batches of the re-encryption mix network, many configurations stipulate that a threshold key is used so that at least a certain number $k$ out of the total number of key shares $n$ must be used to perform the decryption. The key shares can then be distributed among a set of decryption tellers. Unless $k$ of the $n$ tellers work together in the decryption of each encrypted receipt this cannot be done.

It is thus evident that a secure re-encryption mix network is based on a *threshold probabilistic* cryptography scheme. It is “threshold” for the reasons we have just seen, and “probabilistic” because the encryption contains an element of randomness, ensuring that two encryptions of the same plaintext do not necessarily look the same. Two such schemes are Elgamal [26] (Section 2.3.2) and Paillier [65] and implementations of these schemes in Prêt à Voter have been described in [85] and [80] respectively.
2.3. Cryptography

Random partial checks (RPC)

The random partial check is a method that can be used to audit the correct function of a mix network without revealing any complete sequence of links from input to output [43]. Using some method that cannot be controlled or influenced by the mixers, a binary mask is created for the first column of ciphertexts in the mix network, one bit for each of the ciphertexts in this column\(^ {17}\).

A 0 value means that the mixer in question must reveal where the ciphertext originated and what values were recovered (in the case of a decryption mix network) or applied (in the case of a re-encryption mix network) in the process and this data is published. This allows anyone to apply the same process in reverse to the ciphertext and check that the result is the ciphertext in the previous column of the mix network.

Conversely, a 1 value in the mask for a particular ciphertext means that the mixer must reveal the next step in the mix for this particular ciphertext and the associated values. This allows anyone to check that this ciphertext has been properly decrypted (or re-encrypted) and that it can be found in the next column in the mix network.

The process is continued across the whole mix network by propagating the mask such that each ciphertext in the whole mix network has either an incoming or outgoing link and its associated values revealed. Because no ciphertext’s origin and destination are both revealed in any step of the mix network there can be formed no complete link from mix network input to output.

The likelihood that a mixer can cheat “and get away with it” is inversely proportional to the number of ciphertexts in the columns. For a particular ciphertext I, as a mixer, can choose to cheat either in the incoming or the outgoing link — I know that one of these will be audited and so the chance that my cheating will remain unnoticed is 1 in 2. If I cheat with two ciphertexts my chance to get away with this is 1 in 4 as I first have 1 chance in 2 to get away with the first ciphertext and then 1 in 2 to get away with the second. If I cheat with three ciphertexts my chance to get away with this is 1 in 8, a rapidly sinking likelihood.

As random partial checks are computationally relatively inexpensive, they seem particularly appropriate to auditing mix networks used in electronic voting because a cheating mixer must change a certain number of ciphertexts to throw the election (i.e. give the victory to a different candidate than the actual winner). If the number of votes that the cheating mixer has to change is n the likelihood that it gets away with it is one chance in \(2^{-n}\). For \(n = 10\) this is one chance in \(2^{-10} = 1024\), \(\approx 0.09\%\). As the number of changed votes increase the likelihood of avoiding detection falls exponentially.

2.3.7 Zero-knowledge proofs

A zero-knowledge proof is the term for a process whereby an entity A proves to an entity B the possession of some secret, or that something is true, without revealing that secret

\(^{17}\)The mask may be created by some random function performed by a third party (coin tosses) or some pre-agreed method such as the output of a well-known hashing algorithm on the complete mix network data.
or any other information [57]. In an interactive zero-knowledge proof $A$ and $B$ execute a protocol that contains a number of messages exchanged between the two parties and which results in some proof that convinces $B$ of $A$'s possession of the secret. The exact protocol depends on the circumstances: for example $A$ may be proving to $B$ that it knows a secret that $B$ also knows or it may be proving knowledge of the plaintext in a particular ciphertext without $B$ knowing this plaintext.

One example of the use of zero-knowledge proofs in electronic voting systems is the Neff mix [62] where a mixer$^{18}$ can prove that the mix has been performed correctly without revealing any information about the contents of the encrypted votes.

## 2.4 Paper based electronic voting schemes

At first it may seem counter productive to use paper in an electronic voting system, but the motivation for paper in electronic voting is simple: it allows the voter to perform an encryption of her vote using nothing more than her hands. There are different ways of accomplishing this feat and two systems that use paper are Prêt à Voter and Punchscan, both of which are described in this section.

Furthermore, there are obvious benefits of preserving a ballot form that the voters are familiar with, allowing them to fill this out using a method that they are familiar with (namely a pen) and to allow them to do this in the privacy of a voting booth.

Three paper-based voting schemes of the same family (descendants of Chaum’s visual cryptography system) are presented in this section: Prêt à Voter, Punchscan and Scantegrity. The main motivation for the development of all of these schemes has been to provide end-to-end verifiability (Section 2.2.3) and in general this is achieved in two parts. First the voter is given an encrypted receipt that she can use to check the inclusion of her vote in the tally (voter verifiability) and second any interested party is able to check that all those encrypted receipts are properly turned into plaintext votes and that these in turn are tallied correctly.

### 2.4.1 Chaum’s visual cryptography system

The electronic voting system based on a receipt with visual cryptography proposed by David Chaum in 2004 [17] marks the start of an evolution of systems where the voter would create an encrypted receipt of her vote by splitting the ballot form in two$^{19}$ and destroying one half of it. In this scheme the voter interacts with a voting machine to form her vote and a receipt is printed by the machine, detailing the choices that the voter has made. This printout consists of two layers of see-through material onto each of which “half” of the information has been printed such that when light passes through both layers the eye perceives pixels that make out the letters in the names of the candidates for which the voter has cast a vote.

---

$^{18}$Trusted in this case not to break the secrecy of the election

$^{19}$In Chaum’s visual cryptography scheme the printed receipt consists of two layers that are separated; in later systems the ballot form may consist of a single sheet which is cut in two.
2.4. Paper based electronic voting schemes

Figure 2.2: A typical Prêt à Voter ballot form

In fact each pixel that the voter sees in the printout is made up by four “sub-pixels”, two on each layer. When the two layers are separated each appears to have random pixels printed onto it: this is in fact visual cryptography due to Naor and Shamir [60]. However, the seemingly random noise of pixels does in fact encode the contents of the vote (see [17] for details). The voter chooses one of the layers as her receipt and destroys the other. The voting machine submits the same part to a central repository and there it can be decrypted and counted after the close of the election.

The disadvantage of the scheme is that it uses highly specialised printing onto two layers of see-through material, a costly and error prone procedure. But building on the same principles this system spawned two successors: Prêt à Voter and Punchscan, both detailed below.

2.4.2 Prêt à Voter

Introduced by Peter Ryan [76, 81] as a simplification of David Chaum’s early electronic voting scheme [17, 15] and developed [20, 55, 56, 85, 80, 36, 108] and tested [82, 83, 84] in a series of papers, Prêt à Voter [87] is an electronic voting scheme based on a paper ballot form with a candidate list in a random order.

The ballot form

In the typical Prêt à Voter ballot form [76, 20] (example shown in Figure 2.2) the candidates are listed in some order on the left and a grid where the voter is able to mark her choices is on the right. Between these two ballot form columns runs a perforation
along which the voter is able to tear the form in order to detach the two columns from each other. At the bottom right of the form is printed an encrypted value called the onion. Where the name of each candidate appears in the list on the ballot form is determined by the contents of the onion. This varies from one ballot form to another.

**Voting ceremony and verifying the vote**

When the voter has identified herself and been accredited as an eligible voter at the polling station, she is allowed to blindly select a ballot form at random. She keeps it hidden until she reaches the privacy of the voting booth. Here she makes her mark(s) on the ballot form, tears along the perforation and destroys the left hand side carrying the names of the candidates. What remains is the right hand side bearing her mark(s) and the onion. As the candidate list order is different on each form it is impossible to know the contents of the vote from the markings on this right hand side. We therefore call this the encrypted receipt.

When the voter comes out of the voting booth with the encrypted receipt it is scanned and a digital copy is submitted to what is called the web bulletin board. This is a publicly available, append only, web resource. No-one is able to delete information posted to the web bulletin board and only the election authority and trusted parties (see below) are able to post to it. When the encrypted receipt has been scanned the voter is allowed to take it home. Using a computer connected to the world wide web (WWW) she can then look up her receipt on the web bulletin board. The voter can check that the digital representation of the encrypted receipt on the web bulletin board matches her own paper copy, rendering Prêt à Voter voter verifiable (see Section 2.2.3).

In order to guarantee the secrecy of the vote a set of trusted parties, tellers, are used. Tellers should be selected so that as many of the voters as possible have trust in at least one and that the tellers represent such disparate interests that they are very unlikely all to collude. For example a teller could be run by each of the major political parties, another by the United Nations and a third by the European Union. There are two versions of Prêt à Voter that we identify by their years of introduction: 2005 for the decryption mix network configuration and 2006 for the re-encryption mix network configuration.

**Prêt à Voter 2005**

Much like *Allium cepa* (that is to say a common onion) the Prêt à Voter 2005 onion has many layers. In the latter the layers are of encryption in a public key cryptography scheme such as RSA and under each is hidden a nonce value \( g_i \). The sum of the hashes of all nonces under all layers of the onion determines the cyclic shift \( \theta \) of the candidate list, as shown in Equations 2.4 and 2.5.

Each trusted party, teller, has a public key pair and two consecutive layers of the onion are encrypted under this public key, again shown in Equations 2.4 and 2.5. The

\[20\]

Prêt à Voter 2005 refers to the version of the system using decryption mix networks.
decryption of the encrypted receipt into a plaintext vote is performed by a decryption mix network of all tellers.

\[ d_i := \text{hash}(g_i) \mod v \quad i = 0, 1, 2, \ldots, 2k - 1 \]  
\[ \theta := \sum_{i=0}^{2k-1} d_i \mod v \]  

In Prêt à Voter 2005 all ballot forms are prepared and printed by a single election authority and are then distributed to the polling stations [20], giving rise to the authority knowledge, chain of custody and chain voting problems [82, 84, 55, 56] (see Section 2.4.7 for further information). In short all of these problems can be explained by the simple fact that anyone who can see the ballot form before it is turned into an encrypted receipt is able to use the information published on the web bulletin board after the election to learn the contents of a vote without having to perform the otherwise necessary multiple decryptions.

Prêt à Voter 2005 also suffers from some unfortunate problem regarding the robustness of the system. As each layer of the onion is an encryption under a particular teller’s public key only this teller is able to perform the decryption. The onion must thus be decrypted by each teller in strict order. If one of those tellers is unable to perform the decryption, either because of some accident (loss of the private key for example) or because it is performing a denial of service attack, it is impossible to decrypt and tally the election. Furthermore, the decryption mix network is quite powerful in finding any teller that tries to cheat but it offers no method to recover from this.

Prêt à Voter 2006

To address these issues Prêt à Voter 2006 was introduced, using a re-encryption mix network first with Elgamal [85] and subsequently with Paillier [80] encryptions. Both these encryption schemes can be configured to use threshold keys and both are probabilistic, meaning that two encryptions of the same plaintext do not necessarily look the same. Thus it is possible to make the system much more robust and overcome or minimise all of the problems with Prêt à Voter 2005 described above.

To ensure that no-one knows the contents of the ballot form (except the voter) the system uses a combination of distributed creation of the onion and on-demand printing of the ballot form. The process of preparing the ballot form is started by a clerk who sets up two probabilistic encryptions of a known “starting value”: one under the public threshold key of the tellers and one under the public key of a printing clerk (or booth printing device). These two are then passed as a pair from clerk to clerk\(^{21}\), each of which adds further randomness to the plaintext and to the randomness of the encryption.

This re-encryption mix network is supported by the threshold probabilistic encryption scheme used. When the initial clerk sets up the pair of encryptions, onions, they contain the same random plaintext set by this clerk. This represents a candidate list in

\(^{21}\)These entities can be the same as the tellers but as they are not necessarily so, they are named clerks
alphabetical order. When the pair is passed from clerk to clerk each adds randomness to
the plaintext without revealing what randomness they add nor being able to learn the
plaintext, thus mixing the candidate list. This possibility to add to the plaintext is a
feature of the probabilistic encryption schemes Elgamal [26] and Paillier [65] used. When
all clerks have thus contributed to the random plaintext no subset of them can know
the plaintext without first decrypting. The re-encryption mix network is auditable.

The random plaintext of the pair of onions being the basis for the permutation of the
candidate list, the ballot form has been created in a distributed fashion. What remains
before the form can be used to vote is of course to print it onto paper. This is done by
decrypting the onion that has been encrypted under the public key of either the printing
clerk or of a booth device, which reveals the candidate list and allows the ballot form to
be printed. The remaining onion, that encrypted under the public threshold key of the
tellers, is used in the subsequent decryption of the encrypted receipt.

Unfortunately the printing raises a hard problem: how can the ballot form be printed
so that only the voter can know the contents of the ballot form? The form could be
printed on demand in the privacy of the voting booth so that it only exists in its full form
there, but that requires a printing device in the booth that has a private key. Anyone
familiar with private key infrastructures (PKI) understands that this becomes a non-
trivial key distribution problem. If, for example, anyone is able to steal the private key
from within the booth device, or the organisation responsible for loading the key into the
booth device steals it, the secrecy of the election is broken. It may be possible to devise
a scheme whereby the contents of the ballot form is decrypted by a group of trusted
parties, on demand, but again this raises the complexity of the system. Furthermore,
the booth device has to learn the order of the candidate list in order to print it and if it
is able to leak this information then again the secrecy of the election is under threat.

Instead of printing the ballot forms on demand within the voting booth, they can be
printed in advance by some election authority. This again gives rise to the authority
knowledge and chain of custody problems, both of which can be minimised using pro-
cedures. Whether to use the on-demand printing method or the pre-printing method
is the topic of current debate. If there is some technical discovery that can secure the
on-demand printing then this seems to be a very promising method of printing the ballot
form but until then it is likely that the secrecy of the ballot form can more easily be
safeguarded using procedures in the pre-printing chain of custody.

Voting in Prêt à Voter 2006 is done in exactly the same way as in the 2005 version,
but the decryption of the encrypted receipt is quite different. When the voters’ choices
with the associated onions are all published on the web bulletin board after the close
of the election they are then passed between the clerks\(^22\) in their re-encryption mix
network. Note that in this re-encryption mix there is no injection of randomness into
the plaintext, only into the probabilistic part of the encryption.

When a threshold set of the tellers agree that the encrypted receipts have been sa-
tisfactorily shuffled they are able to decrypt the final output batch of the re-encryption
mix network into plaintext votes. It is important that the encryption of the onion has

\(^{22}\text{This can be the same clerks used in the distributed creation of the ballot forms or a different set.}\)
been done under a threshold key and that this key is shared by a set of tellers so that no single entity is able to perform any decryption without the consent of at least a threshold set of all entities working together. This ensures that only the output batch from the re-encryption mix network be decrypted, safeguarding the secrecy of the vote and the privacy of the voters.

2.4.3 Punchscan

Sharing the same mother system in Chaum’s scheme with a visually encrypted receipt [17] it is not surprising that Punchscan [28, 68, 71] and Prêt à Voter have many similarities even though they have had different development paths. Punchscan can be argued to have a distinctly American flavour in the layout of the “native” ballot form on a single sheet of letter size paper but there are also other differences.

Punchscan was originally introduced [68, 28] and subsequently developed [38, 27] by David Chaum and a number of graduate students associated with him. Some analysis of the system has been made by others [55, 56].

The ballot form

As Prêt à Voter has two columns to allow the voter to turn the ballot form into an encrypted receipt (see Section 2.4.2) the ballot form in Punchscan has two pages. Figure 2.3 shows a typical Punchscan ballot form with both its pages overlain. The ballot form has a number of elements. The first is the serial number printed at the top right corner that uniquely identifies the ballot form. As can be seen in Figure 2.4 this is printed identically on both pages of the form. The candidate list is printed on the top sheet in alphabetical order or whatever order is stipulated by electoral law. The candidate list
Chapter 2. State of the art

does not appear on the bottom sheet.

The native Punchscan ballot form\textsuperscript{23} uses indirection to achieve the same effect as the random order candidate list in Prêt à Voter. On the top page of the ballot form, next to each name in the candidate list is printed a random symbol, unique for the candidate on the ballot form but assigned at random on each ballot form. Each symbol can also be seen through the holes punched (hence Punchscan) through the top page, as they are printed on the bottom page, again in a random order from left to right. There is no connection between the symbols assigned to each candidate on the top page and the order in which these symbols appear on the bottom page.

\textbf{Voting ceremony}

When the voter appears at the polling place she identifies herself to a poll station worker and is subsequently allowed to choose a ballot form at random from a pile of ballot forms, hidden from view (laying upside down on the table or in envelopes for example) and proceeds to the voting booth. Using a thick bingo marker (bingo dauber) she marks her choices on the two-page ballot form. This is done by marking the symbol on the bottom page seen through the hole representing the desired candidate. As a thick bingo marker is used a mark is made not only over the symbol on the bottom page but also around the hole punched through the top page, meaning that both pages are marked.

Consider Figure 2.4. When the voter has indicated a vote for Lars the top page has a mark around the fourth hole and the bottom page a mark over the character $b$. When either page is considered without the other it does not indicate the contents of the vote. Thus, when the voter has marked her choice(s) in the voting booth she creates the encrypted receipt by separating the two pages and destroying one. She is free to choose which page to use and which to destroy\textsuperscript{24}. As can be seen in this same figure, the vote is encoded into the position of the mark (numerically if you will, the index number of the position of the mark: 0, 1, 2, 3, ... ) and not into the symbol marked or the correspondence between a candidate and a symbol on the top page.

In order to finally cast the vote the voter approaches poll station workers and allows the encrypted receipt to be scanned and submitted electronically. She is allowed to leave the poll station with the encrypted receipt, enabling her to look this receipt up on a web bulletin board after the close of the election. Punchscan is thus voter verifiable as the voter is able to verify the inclusion of her vote in the final tally.

\textbf{Meeting of officials and the diskless workstation}

In general, it may be argued, Punchscan employs procedural protection against most problems associated with electronic voting systems. To deal with the authority knowledge problem (see Section 2.4.7) for example the system is heavily dependent on procedures.

\textsuperscript{23}We say “native” here because as discussed in Section 4 both Punchscan and Prêt à Voter can use a variety of ballot form configurations, including that of the other sister system.

\textsuperscript{24}Procedural defences against some problems described in Section 2.4.7 may require this choice to be made before the form is filled out.
Figure 2.4: The pages of the Punchscan ballot form
It is well argued by the developers [71, 28, 68, 38, 27] however that procedural defences can be quite effective.

The trust in Punchscan is invested in a number of observers or officials, perhaps nominated by each political party, constituency etc. These have a number of meetings during the running of the election at which all the functions underpinning the electronic voting system are carried out. The meeting of officials is responsible for safeguarding the secrecy of the election while the accuracy of the election is audited by an independent set of auditors. To help them achieve the necessary level of secrecy they use a so-called *diskless workstation*, a computer that does not have any means of communication with the world and thus no way of leaking the secret information.

When the officials come to the meeting they each bring with them the following:

1. A motherboard, graphics card, processor and other components necessary to assemble a working (diskless) computer that they have satisfied themselves do not contain any malicious hardware or software and can thus be trusted not to leak any of the secret information.

2. A copy of the Linux distribution and the necessary software, all of which has been compiled by the official or the organisation he or she represents from code that has been fully analysed.

3. A flash drive/memory stick portable media that can be connected using the USB connector.

As each official has a complete set of the necessary components it is possible to randomly select each component among those available, and thus there is a probability attached to the choosing of a compromised component supplied by an official. Likewise the Linux and software distribution can either simply be selected at random or all those distributions provided can be compared to each other and as they should be exactly identical a distribution is only used if it is exactly what all officials have brought to the meeting.

As the workstation has no hard disk it is booted from the CD holding the Linux distribution. The necessary software is already loaded into this distribution and can accomplish all the functions of the electronic voting system. As the running of the election software creates data that must be published onto a public web bulletin board a memory stick (portable media that connects to the USB port: may have other names) is used and the data is stored onto it. A series of similar sticks may be used to provide backup of the data.

As described in the next section a large amount of secret random data underpins the anonymisation and decryption of the encrypted receipts. The developers have proposed a very interesting procedure whereby no database is created but all the necessary ballot form data is created each time it is needed, i.e. in the meetings of officials. This is

---

25This requires the use of a computer to compute digital hashes on the contents of the read only media on which the operating system copies have been brought.
accomplished by officials entering a series of characters that they have selected autonomously, all these characters being hashed using some publicly available hash function and the resulting hash being used to deterministically create the necessary data. This means that these random character strings, passwords if you will, are all needed to create the data and as it is not stored anywhere it must be recreated using the same passwords each time — making it impossible for any subgroup of officials to create the data. There are also some strategies for making sure that no official is able to stage a denial of service attack by “forgetting” the password [38, 27].

The mixing scheme and its initial audit

Instead of using either a re-encryption (Section 2.3.6) or decryption mix network (Section 2.3.6) a new mixing (anonymising) scheme was introduced with Punchscan and lacking a different term it is here called Punchscanian mix. Its most important difference from the mix networks used in many electronic voting schemes is that it is not based on cryptography as such, except the use of some publicly available hash scheme.

As has been shown in a previous section the vote is encapsulated in the position of the mark on either page of the two-page ballot form and not in the symbol marked on the bottom page or the correspondence between a candidate and a symbol on the top page. The “decryption” of the encrypted receipt in Punchscan is thus the process of figuring out which candidate a particular position corresponds to — or more accurately to move the mark in a series of manipulations such that its final position corresponds to a candidate list in the base (alphabetical) order and can simply be read off.

Originally Punchscan was explained using only two possible manipulations: no change or inversion. While this is the case in a race with exactly two candidates it seems more likely that full permutations must be employed. Therefore here the more general case is presented and each manipulation is the application of a full permutation (it is quite possible to represent the “no change” case as a full permutation and therefore this special case is disregarded).

The decryption of the encrypted receipts is performed using a series of one or more decryption tables (example shown in Figure 2.1) that have seven fields:

1. The input — this must be a full permutation of all the positions to facilitate the application of full permutations, in the first input column of the first of any linked decryption tables this is simply a permutation indicating the base order

2. The first permutation

3. The destination in the middle result where the permuted value is stored

4. A middle result (the first permutation applied to the input)

5. The second permutation

6. The destination in the output where the permuted value is stored
7. The output — a full permutation of all the positions, becomes input to any linked decryption table.

It is an important property of each decryption table that at each application of permutations to the encrypted receipts the latter are randomly mixed. This is quite similar to decryption or re-encryption mix networks where at the removal of a layer of encryption or the re-encryption of the encrypted receipt are these mixed and the addressing in this mix kept secret (until partially audited).

To create the ballot form data the meeting of officials uses the software running on the diskless workstation. Any desired number of chained decryption tables are created and their final result is stored in a printing table and this becomes the basis on which the ballot forms are printed. The printing table must be safeguarded (probably using some procedure) as its transportation to the printing facility, the printing and the transportation of the printed ballot forms to the polling station are all liable to chain of custody problems (Section 2.4.7).

When the printing table has been created and stored onto the movable storage media all the different values in the printing and decryption tables are committed using some public hashing scheme and these commitments are also stored on the media. The printing table must be kept secret and reliably destroyed when the ballot forms have been printed but the commitments to the decryption table values must be reliably published on the web bulletin board so that they are known by all before the start of the election.

As all underlying values have now been committed to the web bulletin board a pre-auditing of some number of all created (but not yet printed) ballot forms is made. The required number of forms (this may be any number) are selected at random or using some deterministic scheme by independent auditors. The selected ballot forms are then...
completely opened, that is to say that the whole chain through the decryption tables(s) and resulting in the printing table is revealed. If there is any malfeasance in the creating of the ballot forms this will be detected by the auditors by some probability that depends on the number of forms that are opened. Note that all values in the opened forms can be checked by first ensuring that the revealed values do indeed correspond to the published hash commitments and that they also result in the committed print table.

As an example of this pre-auditing we can use Table 2.1. Recall that before the pre-audit takes place this table is represented on the web bulletin board simply by a set of hash commitments and all values are thus hidden from view. Suppose form number 3 is selected to be opened. The meeting of officials uses their passwords to recreate the decryption tables\(^\text{26}\) and publishes the information revealed in Table 2.2. Anyone is able to use some software to check that the revealed values in Table 2.2 do correspond to the hashes published in Table 2.1. Note that these hashes must be in some probabilistic scheme so as to prohibit some trivial brute force attack against the decryption table.

In this example it is thus the case that

\[ O_{31} + P_{31} = O_{42} \]

for the first permutation and

\[ O_{42} + P_{42} = O_{23} \]

for the second permutation, arriving at \( O_{23} \) that, if this is the only decryption table, should correspond to the permutation committed in the printing table. The purpose of showing these complete paths through the decryption tables is to, with some probability, attempt to find any ballot forms that have been created erroneously.

### The decryption of the votes

As the consideration here is of Punchscan with full permutations it is necessary to think of the decryption of the encrypted receipts as going from the last output column of the last decryption table to the first input column of the first decryption table. This is because the permutations must be applied in this order to the position of the mark on the encrypted receipt to bring this back to the base order, indicating in plaintext the intention of the voter.

When the encrypted receipts are collected electronically by the election authority they are inserted at the correct place at the end of the particular decryption chain necessary to decrypt that particular vote. When the election has closed a final meeting of officials recreates the decryption data, decrypts all votes and publishes the plaintext votes to the web bulletin board, which can then be tallied by any interested party. At this stage it is important to note that the Punchscarian mix effectively hides the connection between an encrypted receipt and its plaintext vote output after the decryption.

To further audit the decryption data a set of independent auditors now challenges the meeting of officials to reveal exactly half of the data associated with the decryption of the votes. In each decryption table and for each vote either the first permutation and

---

\(\text{26}\) Early on the collected decryption and print tables were called the punchboard by the authors.
shuffle or the second permutation and shuffle are revealed. Each decryption table thus contributes a certain amount of audited data, influencing the probability of finding any error or discrepancy. Thus the larger the number of linked decryption tables used the greater the probability of finding any error\textsuperscript{27}.

It is interesting to note that any number of decryption tables can be linked and the more decryption tables there are, the more data can be audited, influencing the probabilities associated with detecting any errors or foul play in these tables. Furthermore, it is quite easy to imagine that the secrecy of the vote (and the privacy of the voters) can be safeguarded by a number of trusted parties, each providing one or more decryption table(s). In the case where all tables are kept secret by a single meeting of officials this secrecy does definitely depend on this meeting being procedurally prevented from leaking any information that can connect the encrypted receipt to its associated plaintext vote.

In a configuration whereby the encrypted receipts are passed between trusted parties (compare tellers in decryption mix networks, Section 2.3.6) the secrecy of the vote is safeguarded if at least one trusted party is honest.

It is also interesting to note that in the case where a single meeting of officials has custody of all decryption tables these can create any number of new decryption tables that, linked, have exactly the same resulting ballot forms as the previous table(s). These new tables can then be committed before being audited using a challenge from independent auditors. This procedure can be repeated any number of times resulting in more and more data being audited each time and stops when all parties are convinced that the meeting of officials has not committed any errors nor attempted to illegally influence the election. This method allows extensive auditing of the accuracy of the election and does not influence the secrecy of the vote.

\subsection{2.4.4 Scantegrity}

Building on the ideas introduced in the Punchscan system, where no cryptography is necessary except a publicly available hash function, Chaum has introduced the Scantegrity system\textsuperscript{28} as a complement to any optical scan system already available [18]. The purpose of introducing the system has been to add verifiability to these optical scan systems that normally are not voter verifiable. It may be argued, however, that the system is rather cumbersome to challenge, especially to quickly dismiss unfounded challenges, which can easily be made and if made by a large enough number of people will disrupt the election.
2.4. Paper based electronic voting schemes

Figure 2.5: A typical Scantegrity ballot form

The ballot form

An example of a Scantegrity ballot form is shown in Figure 2.5, where the original optical scan ballot form is made up of the candidate list, in the base order stipulated by relevant electoral law, and the “bubbles” that are filled to indicate the candidate to which the vote is awarded. These can confidently be said to be features of most optical scan systems.

What has been added to the ballot form are two things: the chit and the symbols, letters, inside the bubbles. The chit is shown in Figure 2.5 as the top-right corner but could of course be any part of the ballot form. The main requirements on the chit is that it contains the serial number of the ballot form, that the voter can verify that the serial number on the chit corresponds to that on the rest of the ballot form (hence the barcode in this example that stretches across the perforation) and that the chit can be torn off.

As the candidate list is in the base order Scantegrity uses indirection to achieve an encrypted receipt. The symbol associated with each candidate in the list is based on Punchscan decryption tables (see Section 2.4.3) and thus the two systems are compatible.

---

27 Quantifying this probability is hard at this level as it depends on the implementation.
28 The work detailed in this thesis is based on Scantegrity but has at the time of printing been succeeded by Scantegrity II [19].
29 In an optical scan system the voter fills out a plaintext ballot form that is then interpreted by a scanning device, simply by finding the presence of dark ink in a position or the absence of it. In most systems the ballot form is first read and then automatically deposited in a ballot box, making it possible to perform a manual count of the votes.
The voting ceremony and voter verifiability

When the voter enters the polling station and identifies herself to poll station workers she is allowed to take a random ballot form blindly from the collection of forms available. The ballot forms have the same secrecy requirements as the Punchscan ballot forms as anyone able to note down the mapping of symbols to candidates on a particular ballot form is able to check the contents of the vote on the web bulletin board after the close of the election.

In the voting booth the voter marks her choice by filling the bubble next to the preferred candidate with ink, as stipulated by the optical scan system that she may well have used in previous elections. When she is finished she takes a piece of note paper available in the booth and writes down the character shown next to the candidate she is voting for. The voter tears off the chit along its perforation and while she allows the ballot form to be scanned and deposited into the ballot box, she takes the note and the chit home as her encrypted receipt.

After the close of the election the serial numbers of all voted ballot forms together with the symbols of their respective voted candidates are published on the web bulletin board. In order to check that her vote is included in the tally the voter visits the web bulletin board, looks up her serial number and verifies that the symbol shown is that which she has noted down in the voting booth. Note that this does not say how she voted, merely that her vote has been recorded as she cast it.

Challenging the election

If the voter is unable to find her serial number or if the symbol associated with her serial number on the web bulletin board is not the one she has noted down in the voting booth she can decide to challenge the election. In the ideal scenario only voters who feel that they have truly been disenfranchised would challenge the election, but the system must of course handle cases where voters who have other reasons than fairness to put forth a challenge and be able to dispose of such malicious, or simply accidentally erroneous, challenges.

When a challenge is put forward by a voter, which may well be through her political party, directly to the election authority or through the media, the election authority must decide whether or not to allow the challenge to proceed. If so, then the voter (or her representative, in the form of a party official or lawyer for example) is invited to take her chit to a place where original ballot forms collected during election day are held. In this storage the forms are sorted by their serial number, allowing the election authority to find the form in question without problem.

Those present now proceed to check the ballot form and this is done in two steps. To start the first check the election authority finds the voter’s original ballot form\(^\text{30}\) and places it in a special envelope through which only the chit corner is visible, i.e. the place

\(^{30}\)If the ballot form cannot be produced this is evidence that the election authority has not been able to guarantee the integrity of the election — given that it is impossible for a voter to forge a chit. A forged chit that cannot be proven false would seriously compromise the trust in the system.
where the chit has been torn off. The ballot form in its envelope and the chit are then placed under a microscope and it is forensically shown that the chit definitely belongs to the ballot form in the envelope. This is possible because of the naturally present fibres in the paper on which the ballot form is printed, or perhaps by special threads laid into the paper to achieve this purpose [18].

To start the second step of the ballot form check the election authority removes the ballot form from the envelope and places it in another where the candidate list and the associated symbols and bubbles are shown, but this is done face down so that no person present is able to see the contents of the vote. This envelope is then placed in a container with a number of similar envelopes showing ballot forms, these are thoroughly mixed and for all the ballot forms now in this container the following are true:

1. There is one vote for each candidate in the container
2. Each ballot form in the container has the same symbol marked

All the ballot forms in their envelopes are now taken out of the container and placed on the table and it is shown that even though they are all votes for different candidates they are all marked with the same symbol. All the forms on the table can be examined in detail to show that the position of their marks have not been changed at any time.

Although this may not be fully intuitive it is now the case that the election authority has shown the voter that her ballot form is among those on the table and that all those on the table have the same symbol marked. If this symbol does not correspond to the symbol the voter has noted down in the voting booth it has been proven that the voter has an incorrect recollection of the symbol she has marked. Note that the candidate for which the voter has cast her vote has not been revealed to any person in the room, although the election authority is quite aware of it.

Discussion

The main achievement of Scantegrity is that it adds voter verifiability to already existing optical scanning voting systems. However, it does have two arguably severe drawbacks:

1. The election authority knows the votes of each voter, or invariably learns the vote of a voter who challenges the election.
2. The two-step challenge process with its envelopes and shuffling has an unfortunate air of a magic trick and it is arguably quite possible for the voter to feel that she is being deceived by sleight of hand.

2.4.5 Scantegrity II

With the view to make the verifiability and challenging processes of Scantegrity more practical, Scantegrity II\[31\] [19] has replaced the necessity to forensically match a chit to a ballot form with a clever way of using invisible ink.

\[31\]“II” both denotes that the system is the second with the same name and that the system uses Invisible Ink.
The ballot form

Similar to the original scheme, the Scantegrity II ballot form is an adapted optical scan ballot form. The voter makes a choice by filling a bubble next to the desired candidate’s name using a pen. Also like in the original scheme, this bubble contains a confirmation value — but in this scheme it has been printed in invisible ink. Each bubble on the ballot form contains a two character confirmation code that is invisible to the naked eye\(^{32}\) but is revealed by the voter when she fills out the bubble to indicate her vote. The bubble becomes dark when it is covered with a special pen and the confirmation code is revealed as light characters within this dark bubble. This means that optical scan machines will detect the mark and the voter sees the confirmation code.

The ballot form is similar to that in the original scheme, with a serial number printed on the top. The bottom part of the ballot form is reserved for the voter to note down the confirmation codes associated with the choices she makes on her ballot form\(^{33}\). The bottom part is used so as to cause as few problems as possible when the sheet is fed through the scanner.

The voting ceremony

The voter enters the polling station, identifies herself and is checked against the register of eligible voters. If she is eligible to vote she is allowed to take a ballot form at random from a pile. Note that the ballot form is not secret (as it was in the previous scheme) so there is no need to hide it at this stage. She proceeds into a polling booth where she fills out the ballot form using a special pen that develops the invisible ink in the bubbles, both revealing the confirmation code within and marking the bubble.

When she is finished she notes down the confirmation codes associated with each of her choices in the space on the ballot form reserved for this and exits the voting booth. The part of the ballot form that holds the notes made by the voter also states the serial number of the ballot form. This part is detached and when the ballot form has been scanned an election official stamps it to indicate that it is a valid receipt.

Publishing auditing information

When the ballot form is scanned, the scanner reads only the serial number (which is encoded in some form that the scanner can read, such as a barcode) and the position of the marks on the form — in exactly the same way that it would read a form that did not have the Scantegrity confirmation codes printed on it. This data is stored in a database and tallied just as in previous elections. After the election the data in the database is used to generate the confirmation codes associated with this form and these are published on the web alongside the ballot form’s serial number (see [19] for details on this procedure).

\(^{32}\)A beautiful effect of this is that the ballot form is not secret before it has been used to cast a vote: it is impossible for anyone to see and note down the confirmation codes associated with candidates on the ballot form as they are invisible.

\(^{33}\)This American style ballot form has a number of races on the same sheet.
Auditing a vote and resolving disputes

A voter who wishes to verify her vote visits the official election website and enters her ballot form serial number, which is printed on the piece of the ballot form that also bears her confirmation code notes and that she has kept. This brings up a list of the confirmation codes associated with the voter’s ballot form and she is able to check this against the notes she made at the time of casting the vote. If these match up she can be satisfied that her vote has been scanned, interpreted and tallied correctly.

In the original Scantegrity scheme any dispute would have to be resolved in a cumbersome procedure involving a chit matched to a paper ballot form. In Scantegrity II the voter can only know the confirmation code associated with the candidate she voted for. Therefore, the knowledge of a valid confirmation code for the ballot form in question is all the proof the voter needs to challenge the election. It is then possible to show, without consulting the physical ballot form (which is kept in an archive) if the confirmation code is valid and if it is the confirmation code that has been aggregated by the election authority.

Discussion

The secrecy guarantees of Scantegrity II only holds if the invisible ink is truly invisible: it must not be visible when the voter holds the form up against the light, nor must there exist a way for anyone to see the confirmation codes without spoiling the ballot form before it is used to cast a vote. Furthermore, if anyone is able to inspect the ballot form after it has been used to cast a vote then this person is able to see what vote is associated with that particular ballot form serial number. This does not associate the vote with a voter but in a scenario where it could be associated with the voter this breaks the secrecy of the election. A special mix of invisible ink that is only visible for a short period of time, enough for the voter to note down the confirmation code, has been envisaged as a solution to this problem [19].

2.4.6 Scratch & Vote

Scratch & Vote [4] uses the normal ballot form and homomorphic tallying in the back-end to simplify the breaking of the link between encrypted receipt and plaintext vote.

The ballot form

The ballot form in Scratch & Vote is the same as in Prêt à Voter: on the left there is a random-order candidate list and on the right a grid where the voter makes her marks. On the right-hand side is, like in Prêt à Voter, an onion value contained in a 2D barcode, but there is also a scratch strip covering an auditing value that the voter can scratch off if she wishes to audit the ballot form. A ballot form where this value has been uncovered

34This is dependent on public verifiability of the system’s back-end, not described here.
35The authors show how it can be used both with the Prêt à Voter and Punchscan ballot forms.
cannot be used to cast a vote\textsuperscript{36}. The part of the ballot form that is covered with the scratch strip has also got a perforation around it, making it possible to detach this part from the ballot form.

**The onion**

The onion, printed as a 2D barcode on the ballot form, encrypts the candidate list order as a series of Paillier\textsuperscript{[65]} ciphertexts. This cryptography scheme is such that the plaintext of two ciphertexts, encrypted under the same key, can be added together without first decrypting the values. Scratch & Vote makes use of this by encoding the identity of each candidate as $2^{M(i-1)}$ where $2^M$ is the number of votes the system can handle and $i$ is the index of the candidate in the canonical ordering. This is best illustrated by an example\textsuperscript{37} ($M = 28$):

<table>
<thead>
<tr>
<th>$i$</th>
<th>Candidate</th>
<th>$2^{M(i-1)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Adam</td>
<td>$2^0$</td>
</tr>
<tr>
<td>2</td>
<td>Bob</td>
<td>$2^{28}$</td>
</tr>
<tr>
<td>3</td>
<td>Charlie</td>
<td>$2^{56}$</td>
</tr>
<tr>
<td>4</td>
<td>David</td>
<td>$2^{84}$</td>
</tr>
</tbody>
</table>

Encrypting the value that identifies a candidate in the probabilistic Paillier encryption scheme renders an encryption of the form $E_{PK_T}(2^{M(i-1)}, r)$ where $PK_T$ is the public key of the tellers and $r$ is a random value. Because the scheme is probabilistic (that is to say that it incorporates this random value) multiple encryptions of the same plaintext do not (necessarily) result in the same ciphertext.

Scratch & Vote takes advantage of this by forming the onion simply as a list of encryptions of the identities of the candidates in the order that they are printed in the candidate list on the ballot form. Continuing the previous example, a ballot form with the following random candidate list order has the following values in the onion:

<table>
<thead>
<tr>
<th>Candidate</th>
<th>In the onion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bob</td>
<td>$E_{PK_T}(2^{28}; r_1)$</td>
</tr>
<tr>
<td>Charlie</td>
<td>$E_{PK_T}(2^{56}; r_2)$</td>
</tr>
<tr>
<td>David</td>
<td>$E_{PK_T}(2^{84}; r_3)$</td>
</tr>
<tr>
<td>Adam</td>
<td>$E_{PK_T}(2^{0}; r_4)$</td>
</tr>
</tbody>
</table>

**The voting ceremony**

Voting takes place just as in Prêt à Voter described above. When the voter exits the polling booth she has destroyed the left-hand half of the ballot form and she is carrying the encrypted receipt. She approaches election officials who check that the scratch strip is intact before tearing along the perforation, separating the scratch strip from the rest

\textsuperscript{36}This is enforced procedurally by election officials.

\textsuperscript{37}This example is from [4].
of the encrypted receipt. The scratch strip must be destroyed to ensure that its values cannot be known by anyone who knows which ballot form it corresponds to.

The remaining encrypted receipt is scanned and published on the web bulletin board. The voter takes the receipt home with her and is able to use this to check for the inclusion of her vote in the final tally (in exactly the same way as in Prêt à Voter).

**Auditing a ballot form**

Under the scratch strip the randomisation values \((r_1, r_2\) etc in the example above) that have been used in the encryptions of the candidates in the onion is printed. In order to audit the ballot form, which the voter is free to do for any number of ballot forms before she casts her ballot, the voter scratches off the scratch strip and approaches staff from a “helper organisation” in the polling station. These use a computer to check that the onion is correctly formed and corresponds to the candidate list as it is printed on the ballot form. This auditing method means that no trusted parties (tellers) must be online to perform the audit of a ballot form (as they do in Prêt à Voter) and the voter is free to choose any helper organisation to perform the audit.

As the public key \(PK_T\) of the tellers and the \(M\) and \(i\) values are published before the start of the election it is trivial for the helper organisation to audit a ballot form for which the scratch strip has been scratched off. For the ballot form in the example above, the top candidate is Bob and the onion value that corresponds to this position in the candidate list is \(E_{PK_T}(2^{28}; r_1)\). Looking up the published value for Bob, \(2^{28}\), the public key of the tellers, \(PK_T\), and the randomisation value printed under the scratch strip, \(r_1\), the helper organisation simply performs the Paillier encryption and compares the resulting ciphertext to that in the onion printed on the ballot form.

**Tabulation**

After the close of the election all the encrypted receipts that have been cast are published on the web bulletin board. The ciphertext for the voted candidate on each of these ballot forms is now extracted. In the example above, if the voter has marked David as her favourite choice\(^{38}\) then the encrypted value extracted from this ballot form is \(E_{PK_T}(2^{84}; r_3)\). In fact the example ballot form would be in the following form:

<table>
<thead>
<tr>
<th>Candidate</th>
<th>Choice</th>
</tr>
</thead>
<tbody>
<tr>
<td>(E_{PK_T}(2^{28}; r_1))</td>
<td></td>
</tr>
<tr>
<td>(E_{PK_T}(2^{56}; r_2))</td>
<td></td>
</tr>
<tr>
<td>(E_{PK_T}(2^{84}; r_3))</td>
<td>X</td>
</tr>
<tr>
<td>(E_{PK_T}(2^{20}; r_4))</td>
<td></td>
</tr>
</tbody>
</table>

As the candidate identities and the randomisation values are unknown and all we have is the encryption of these values, it is not possible to know, without decrypting, what candidate the vote is for.

\(^{38}\)Which we would encourage...
The ciphertexts representing the chosen candidates on all ballot forms are aggregated. This is a fully, publicly verifiable process as the encrypted receipts are published online and anyone can make, and check, the extraction of the ciphertexts. These ciphertexts are then all added together, using the homomorphic property of the cryptography scheme, to form a single ciphertext that contains all the votes. When all tellers agree that this has been done correctly and that all votes have been added to the total, the single ciphertext is decrypted by a threshold set of tellers, revealing the result of the election. Various zero knowledge proofs are published along with this decryption to show that the decryption has been done correctly.

Discussion

The homomorphic tabulation of the result of the election means that no mixing has to take place and therefore this tabulation process should be faster (as it involves less work performed by various parties) whilst remaining fully verifiable. However, it is not possible to support STV or any other multiple-choice election method in this summation. Furthermore, the ballot form, as in Prêt à Voter and Punchscan, remains secret in that anyone who is able to see and note down its contents before it is turned into an encrypted receipt is able to circumvent the cryptography and see the contents of the vote after the close of the election.

2.4.7 Problems in paper-based electronic voting schemes

This section introduces known problems in paper-based electronic voting schemes. These are not necessarily problems that stop the use of electronic voting schemes nor make them any less secure — in most cases the problems are simply considerations that system designers must be aware of when constructing their schemes.

Authority knowledge

The authority knowledge problem is not necessary unique to electronic voting but it certainly is a problem that has thrown a spanner in the works of many otherwise quite secure electronic voting systems. The problem occurs whenever the secrecy (or the accuracy or privacy) of the system depends on some secret that the election authority either creates or is able to learn at some point. This may pose a serious threat to the democratic process if the election authority leaks information that compromises the secrecy of the votes. Systems developers have chosen different ways of dealing with this problem:

- Make no change — Motivated by the argument that the election authority should be trustworthy and can be depended upon not to leak this information. May be acceptable in stable democracies but it seems that electronic voting should make it impossible for an intimidating, hostile government to coerce voters.\(^3^9\)

\(^3^9\)This is merely a reflection and not within the scope of this thesis.
• Add procedures — The motivation here is that procedures can be put in place to
minimise the risk of information being leaked. For example, if the ballot forms are
prepared and printed by the election authority this might be done on a “diskless
workstation” that has been put together by independently verified components
from a number of different sources and booted from a randomly selected Linux
distribution, all in the presence of observers from all participating political parties.
After the ballot forms have been printed the underlying information is destroyed.

• Distributed creation and on-demand printing — It has been shown how proba-
bilistic encryption schemes in re-encryption mix networks (see Section 2.3.6) can
be used to distributedly create the secret information needed resulting in a secret
that can only be “broken” if a number of parties with quite disparate interests
collude, providing safety in numbers. This secret is kept in its encrypted form
until it is printed within the booth in front of an individual voter. Unfortunately
this solution to the authority knowledge problem requires a substantial amount of
hardware: each polling station must have at least one ballot form printing device.
There are further problems described previously.

Chain of custody

Quite similar to the authority knowledge problem the chain of custody problem exists
when the secrecy of the election is based on a ballot form with secret content. However,
traditional paper-based schemes can in fact be said to be chain of custody systems [3]
because correct ballot forms must be printed and transported to polling stations, given
to voters, collected again, transported to counting facilities and finally counted properly
— all safeguarded by procedures rather than technology.

In electronic voting systems the main issue referenced as the chain of custody problem
is the transportation of pre-printed ballot forms from the printing facility to the polling
station. Anyone who is able to see the ballot form before they are turned into encrypted
receipts can circumvent all cryptography and find out the contents of a vote from the
information published after the end of the election. A number of procedural solutions to
this problem have been proposed, including the printing of a lottery-type scratch strip
over the ballot form identifier [82]. On-demand printing is also a suggested solution [85]
to the problem.

Chain voting

With an origin in traditional paper-based voting schemes, the chain voting attack is also
a problem that has the potential to be significantly magnified in an electronic voting
system. The traditional attack is possible in an election system where the ballot form
is a restricted resource. As an example let us assume that when the voter has identified
herself to poll station workers she is given exactly one ballot form. She is allowed to
go into the voting booth and when she emerges poll station workers are attempting to
make sure that she casts the form into the ballot box. Normally it is thus hard to leave
the polling station with a ballot form.
The coercer who wishes to employ the chain voting attack must somehow smuggle a valid ballot form out of the polling station, perhaps by posting some other piece of paper into the ballot box. The coercer then fills out the ballot form to form a vote for his preferred candidate(s) and approaches a voter. “If you use this ballot form and come out with a new, blank one,” he says, “I will reward you.” The coerced voter goes into the polling station, identifies herself, is given a ballot form and goes into the voting booth. Here she takes out the ballot form given to her by the coercer and puts the new, blank form in her pocket. When she emerges from the voting booth polling station workers ensure she posts a ballot form in the ballot box.

When the voter comes out from the polling station with a blank form the coercer can be quite sure that she has cast the ballot form he had given her in advance and he thus rewards the voter. The voter may have spoilt the vote by drawing on the form but in all likelihood she has at least been prevented from voting in accordance with her own choices. When the coercer fills out this new blank form in the same fashion as previously and approaches a new voter with the same offer, he has created a chain.

Consider electronic voting schemes that issue an encrypted receipt. If the voter is able to smuggle a ballot form or an encrypted receipt out of the polling station he creates an encrypted receipt of a vote for his preferred candidate(s) and gives it to the voter, noting down the serial number and the marks on the receipt. Because of the voter verifiability property of the electronic voting scheme the coercer can then check after the close of the election that the voter has properly submitted his receipt as her vote. Previously we saw that the voter could have spoilt the vote but in this case the coercer is given the unfortunate power to check that the voter has complied.

Proposed solutions to the chain voting attack are mainly procedural:

- When the voter is given the ballot form the serial number is noted. When she subsequently casts her vote it is checked that she is using the same ballot form.

- Ballot forms are locked into a large clipboard that is very hard to smuggle out of the polling station. At the time of casting the vote poll station workers check that the form has not been removed from the clipboard, which would be indicated by a torn corner on the form.

Randomisation attack

The randomisation attack is generally understood to be where a coercer entices a voter to vote in a way that results in a receipt that looks in a specific way. For example, in Prêt à Voter this may be to vote only for the candidate at the top of the form, resulting in an encrypted receipt with a mark in the top position and no other marks. The coercer has no way of knowing which candidate the voter has voted for but as the ballot form is allocated to the voter randomly then the vote will also be for a random candidate.

A randomisation attack is thus where a vote is changed to a vote for a random candidate. If a certain number of votes are randomly (re)distributed this would benefit a

\[40\text{This idea was presented by David Chaum at WOTE 2007 in Ottawa, Canada.}\]
candidate with a small number of original votes and certainly have a negative effect on the number of votes given to candidates with a majority of the original votes.

In some electronic voting systems, particularly direct recording devices (DRE), the voter interacts with a machine and tells it how she wants to vote. As the machine learns the contents of the vote any malicious code running on that machine has an incentive to try to change the vote. In other electronic voting schemes the machine never has the opportunity to learn what candidate(s) the vote is for and even if it is able to change the contents of the vote it cannot know what candidate it is removing a vote from and to which candidate it is giving it. This is arguably a variant of a randomisation attack.

**Denial of service attack**

A denial of service attack can either be when someone is stopping a service provider from providing its service or when a service provider is refusing to provide that service. Systems distributed across the Internet are particularly vulnerable as in this context the denial of service attack is most frequently constituted by a very large number of requests being directed at one or more servers, drowning them in false requests and stopping them from providing their service.

Some electronic voting schemes are vulnerable to denial of service attacks, in the first instance because some systems are connected to the Internet. However, some systems (for example Prêt à Voter, see Section 2.4.2) use a series of trusted parties that all must provide a decryption service for the votes to be decrypted and become talliable. Consider the case where such a trusted party plays along all the time through to the decryption phase of the election and then simply refuses to do its job correctly.

There are many plausible solutions to the denial of service attack. For example, a trusted party may in reality be made up of three different organisations working in concert under a threshold cryptography scheme. If one of these is unable to perform its duty because of some accident or because it is performing a denial of service attack then the two remaining can perform the necessary work without this lost party. A web bulletin board is a particularly vulnerable Internet connected entity in the electronic voting scheme and may be made robust using several boards using peer-to-peer technology to share all published information quickly and efficiently.

**The Italian attack**

Although the origin of its name remains undetermined, the Italian attack [36, 108] is possible in most election systems where the voter marks more than one choice. In order to preserve vote and voter anonymity, most voting systems stipulate that a vote is spoilt if any other mark is made on the form than the mark(s) required to indicate for which candidate(s) the vote is cast. If this was not the case, a coercer might ask a voter to sign his or her name on her ballot form, enabling the coercer to see the content of that particular voter’s vote when the votes are subsequently counted in public. However, there may be other ways in which the voter may “sign” a ballot form such that the only one who can see this “signature” is the coercer.
The Italian attack aims to “sign” a ballot form by repeating an unlikely voting pattern, predetermined by the coercer, on a multi-choice ballot form. For example, in a Single Transferable Vote (STV) election with many candidates and/or many races, the number of possible ways a vote can be configured may be very large — enough so to allow a coercer to assign unique permutations to quite a large number of coerced voters. When the votes are subsequently counted in public, or, as in some places, even published, the coercer can identify the ballot form assigned to a particular voter.

The coercer constructs the unique permutation given to a voter by first selecting the candidate the coercer wishes to win and thereafter appending a unique series of choices. In some instances, such as proposals by [36] and [108], the full permutation may not be revealed for a ballot form and the coercer then selects a different strategy. Here the coercer selects a unique permutation of candidates very likely to be eliminated first and appends the candidate that he wishes to win. As the vote goes through the various transfers more and more of these choices are revealed until the vote finally counts for the coercer’s preferred candidate [90]. If it is possible to determine between what series of candidates a particular vote is transferred, the signature permutation emerges and the identification is trivial. However, although no proof is offered for this statement, it intuitively seems that it may be possible to construct the permutation such that even a statistical analysis of the transfers at various stages of the tallying process may reveal the identity of a ballot form.

Everlasting privacy

One major theme in this thesis is the balancing act in the marriage of vote secrecy and verifiability. Although it is important to be able to show at the time of the election that the cryptography used in an electronic voting scheme is secure and thus ensures that votes are not changed and that the anonymity of the votes is not threatened, this is likely not enough to instill complete trust in voters and observers. The reason for this is that some proof of the correct operation of the electronic voting system has to be published and if the cryptography scheme in question had been broken, this proof would either be meaningless and/or seriously threaten both the secrecy and the accuracy of the election.

To illustrate the point, imagine an end-to-end verifiable electronic voting system that might have been introduced ten years ago. Although this hypothetical system employed state of the art cryptography, there may be “chinks in the armour” of that cryptography scheme emerging among scientists. Now imagine that such a verifiable system was used in an election 20 years ago and data about the election published at the time. (This data of course replicated and stored in various repositories both by the state, observers and other organisations.) It is now becoming quite possible that the scheme does not withstand modern cryptanalysis or attacks by modern computer systems. Go back further, and key lengths likely in use at that time soon become trivial for modern machines to break.

Now consider an end-to-end verifiable electronic voting system introduced and used in an election today. It may be based on state of the art cryptography and use the longest possible keys supported by modern computing, but it is impossible to say if,
and if so when, the particular cryptographic scheme may be broken. In fact, it is not impossible that the scheme is already broken: in cryptography we often have to rely on probabilities. If it is secure today, then how long will it take before we consider it insecure?

If we start by imagining something quite probable such as the phrase “the cryptography scheme and/or its keylength will be broken in 100 years” this implies that the accuracy\(^{41}\) (that is to say how hard it is for someone to change votes without anyone noticing) and secrecy of the electronic election will be completely ruined in 100 years. This may not mean much to the candidates and political parties because they have been (correctly) awarded their seats in the present. But many voters might worry about what their grandchildren will think about them when they find out how they voted. In most cases this probably does not deter a voter from following his or her own will, but in some instances this may start to influence the group of candidates that the voter is comfortable voting for\(^{42}\).

Now imagine that cryptanalysis is very fast these days and that may halve the expected life span of the cryptography scheme in question to 50 years. This then means that the voter’s children will find out and that the voter herself may still be alive. Then imagine that as the development of normal computers keeps going forward and faster and faster machines are commonly available, the expected lifetime may again be halved to 25 years. With supercomputing facilities available to states and many organisations, we may have to limit the number of years we can hope for the scheme to be secure to 10 years. Then imagine that cryptanalysis on quantum computers soon may become available and further reduce the effort needed to break the code.

We have now illustrated that voters may worry that even if the secrecy of the election is guaranteed today, it may be broken tomorrow. This means that electronic voting systems may have to be constructed in such a way as not to base the secrecy of the election on the “unbreakability” of a cryptography scheme but on some other condition that will (provably) never be broken. This is the concept of everlasting privacy introduced and addressed by [59].

\(^{41}\)Although it is rather too late to change the outcome of the election...

\(^{42}\)This is only the case if encrypted votes are recorded against the names of voters in the published election data. However, a coercer may require a photocopy of an encrypted receipt from the voter and tell her that he will keep it until the cryptography is eventually broken.
3 An implementation of Prêt à Voter

Building an implementation of an electronic voting system that has matured as much as Prêt à Voter is important to test assumptions and scalability among other properties of the system. This chapter is an attempt at passing on a large set of experiences from this work ranging from the theoretical to the practical.

It is well known that implementing security mechanisms is very hard and our experience do not deviate: when we started building the first implementation of Prêt à Voter we had seen few implementations of paper-based, mixnet style electronic voting systems and so much effort at the start of the project was spent on the translation of theory to practice.

One issue that we focused on solving in this work has been that of implementing Single Transferable Vote (STV). Implementing STV in Prêt à Voter requires that the candidate list be completely permuted (not merely cyclicly shifted) and at the time of this work there was no known setup of Prêt à Voter in which full permutations was supported in re-encryption mixes\(^1\) (Section 2.3.6) and we were therefore forced to implement the system using decryption mixes (Section 2.3.6). The contribution of this research has also been in meeting various other quite difficult real-world election requirements: concurrent polling stations, keeping a full paper trail, laying out a ballot form paper with a large number of races and candidates and more, detailed in this chapter.

3.1 Introduction

The first version of the Prêt à Voter implementation was made specifically to enter the International Student Electronic Voting Systems Competition (VoComp) [93] held in Portland, Oregon in 2007.

The requirements for entry into VoComp remained in draft for several months and all through the competition. However, it was clear that the objective for running the competition was to encourage students and researchers in the electronic voting field to make implementations of the systems that they work on. Four teams were selected and took part in the final in Portland. It was made clear by the judges that their final decisions were very hard to reach and although our team did not win the award for overall best system, we did receive two others: “runner up” and Best System Design.

As part of a VoComp entry a team should try to run a (binding) student election, or if necessary, some other election to test the system. Submission of papers detailing aspects of the own system or analyses of competing systems were encouraged. The format of the final itself was as follows:

\(^1\)After this work several papers has been published on this topic, such as [36] and [108].
1. Each team held a presentation of the own system and its strengths.

2. A demo session where judges and independent attendees were allowed to cast votes and fill out questionnaires.

3. For each system all other teams presented a ten minute analysis after which the defending team was given a 20 minute opportunity for rebuttal.

There were also invited and contributed talks given over the course of the three days. The Prêt à Voter team was a joint team between University of Surrey and University of Newcastle upon Tyne. From Surrey students (Joson) Zhe Xia, Phil Howard and David Bismark (then Lundin) and advisors James Heather, Roger Peel and Steve Schneider joined student Kieran Leach and advisor Peter Ryan from Newcastle. The development meetings were held at Surrey and although some advice did come from Newcastle all the code was eventually written in Surrey.

Another section in this chapter gives detail on the problems encountered and the lessons learned from creating the implementation but it should be noted here that development meetings were held weekly from October 2006 through to February 2007. As this work progressed the team initiated discussions with the University of Surrey Students’ Union (USSU) and it was negotiated that the Sabbatical elections 2007 would be run on Prêt à Voter.

After the opening of the election and the casting of a few dozen ballots the decision was taken by USSU to cancel the electronic voting and revert to a paper election. The team discussed this with USSU but although it was unclear to the electronic voting team the reasons behind the decision, the casting of electronic votes was discontinued.

USSU officials offered to use the pre-printed Prêt à Voter ballot forms to cast their votes and the teams was given three hours the morning after the close of the election to scan all these, which was also done. One property of the Prêt à Voter system is that it asks the voter to clarify an encrypted receipt that it cannot understand or that does not constitute a valid vote. The lack of these checks at the time of voting, when any mistakes could be rectified by the voter, made it nearly impossible to make any useful inferences from the scanned ballot forms. This was not necessarily because voters had spoilt their votes but also because they did not adhere to the necessary marking requirements under which the optical character recognition (OCR) would work.

### 3.2 Requirements

Although not part of VoComp, there were three important special requirements posed by USSU:

1. Single Transferable Vote (STV) — The USSU elections use STV which means that each voter ranks one or more candidates on the ballot form in the order they would prefer them. In a number of rounds during the tally the weakest candidate is eliminated and the votes distributed to the next choice in the list until a candidate has gained a majority of the votes.
2. Paper trail — USSU asked us to allow the original ballot forms to be placed in a sealed ballot box so that they may be counted by hand after the election. This was a requirement from the National Union of Students (NUS) to allow the electronic voting trial to go ahead at the University of Surrey.

3. Remote voting via web interface — USSU was keen on using the electronic election to encourage greater participation in the democratic process among the electorate. After initially wishing for all voting to be done through a web interface they settled on a requirement that those students on placement or otherwise unable to vote in polling stations would be allowed to apply to do so remotely.

3.3 The system

The development of the system was driven by the requirements explained in Section 3.2. In a reiterative fashion the group discussed the requirements and possible solutions to these (which included inventions and the use of techniques already described theoretically) and the consequences of using any of these solutions. A discussion of the lessons learned and the compromises necessary can be found in Section 3.5.

It was decided by the group that the nature of the system is such that all code written as part of the project should be released under a suitable open source licence — the release of code was also a requirement for entry to VoComp. Although some parts of the system must be implementable in any programming language (for example in order to be able to audit the mix network) this open source nature of the system indicated that Java would be a suitable common programming language. As this was the language most team members were familiar with it was decided that Java would form the basis of the system. However, the feeling was that PHP supported by a MySQL database would be more suitable to the simple serving of pages on a web server — both of these are also open source projects.

3.3.1 The Java classes

During the first phase of the project, as the discussions on technical solutions where only theoretical solutions existed were going on, work was done to specify the system components. As all entities in the system were to use Java it was only natural that the ballot form would be encapsulated in a Java class hierarchy. A UML diagram of this class hierarchy is shown in Figure 3.1. The requirements driving the development of this hierarchy were:

1. Each printed ballot form has a single serial number (and barcode) but there may be more than one race on the ballot form. This was represented by a ballot form paper class which can contain any number of ballot forms.

2. Each ballot form has two parts: a candidate list (used in the creation of the ballot forms) and a voting form where the voter’s choice(s) are stored.
3. The candidate list and the voters' choice(s) are both represented by a complete permutation: the previous as a permutation of the index numbers from 0 (the \textit{Permutation} object) and the latter as a permutation of the symbols in question (the \textit{CharacterPermutation} object).

\subsection{3.3.2 Networking}

It was important to immediately implement support for the distributed mixing in Prêt à Voter in the form of networking capabilities between the Java classes. The group decided to use the native Java Remote Method Invocation (RMI) functionality. This would allow the system to be set up on one machine or to be distributed over a network. In Figure 3.2 is shown the following:

- The Web Bulletin Board acts as a server in most communication (except when it itself connects to the Teller). The Web Bulletin Board has a database from which it serves data as required to other components of the system and into which it writes all data collected from the system.

- The Registry module allows a poll station worker to look up a voter in the table of registered voters and this is done by the Registry module submitting a \texttt{voterLookup} RMI call to the Web Bulletin Board which performs the search in its database and returns the data necessary for the poll station worker to verify the identity of the voter in the form of a \texttt{Voter} object. When this lookup has been done, the poll station worker can either assign a ballot form to this voter or cancel a form previously assigned. These operations are \texttt{assignFormToVoter} and \texttt{cancelFormForVoter} respectively. Assigning a ballot form to a voter$^2$ primes this ballot form for submission in the database. The Web Bulletin Board will therefore later accept a submission of this ballot form from the Booth module. If a voter reveals the ballot form or makes a mistake, the ballot form can be cancelled, which allows another form to be assigned to this voter. The Web Bulletin Board keeps a record of all ballot forms in its database, providing a full audit trail of ballot form assignments and cancellations. It goes without saying that a ballot form cannot be assigned to more than one voter and that each voter can only be assigned one ballot form.

- The Booth and Audit Machine modules are principally the same in that they both run on a computer connected to the scanner(s) which reads in the right-hand side of ballot forms. If the ballot form as submitted contains any voter-made mark the Booth module is invoked and asked to submit the vote to the Web Bulletin Board (\texttt{postBallot}), which it attempts in all cases except if the scanning and OCR processes have failed, in which case it cancels the ballot form (\texttt{cancelForm}). If the scanned ballot form contains no mark, this is treated as a ballot form audit and

---

$^2$Keep in mind that this assignment is done on ballot form serial number and a property of Prêt à Voter is that this does not leak the vote.
Figure 3.1: Ballot form class diagram
Chapter 3. An implementation of Prêt à Voter

the serial number of the ballot form is submitted by the Audit Machine module to the Web Bulletin Board (auditForm). This causes the Web Bulletin Board to call each Teller in turn (audit) to reveal the values encrypted into the ballot form onion and then return a Receipt object to the Audit Machine, which contains the ordering of the candidates on the ballot form. This is printed and the voter is able to compare the ballot form as it was given to her with the ballot form printed. All the primitives etc required to truly verify the correct construction of the ballot form are stored in the database and immediately available via the Website.

- The Remote Booth module builds on the Booth module and is used to connect the Remote Voting Applet to the Web Bulletin Board.

- The Teller acts exclusively as a server and its duty is to implement a series of secrecy and security requirements put in place by the organisation running a particular Teller to ensure that the secrecy and integrity of the election remains intact. The Teller must therefore, for example, keep track of which ballot forms it has agreed to audit during the election phase and subsequently not allow these to be decrypted as part of the result tallying. After the close of the election the Web Bulletin Board is instructed to submit a batch of votes to the Teller (processVote) which returns a once decrypted and shuffled batch of encrypted votes. These are stored in the database and the batch is passed to the next Teller. The return batch from the final teller contains the plaintext votes. When this has been done the Web Bulletin Board audits each batch at each Teller by asking it to reveal a set of links (reveal).

- The Administrator module is only run once and this is done before the start of the election. The module is started on a so called diskless workstation, that is to say a computer which lacks a hard drive or any other method of storing data, and used to construct the ballot forms. The ballot form print run is output to a printer, as seen in the UML diagram, and SQL inserts for the non-secret data are created and stored on a memory stick. This is done in the presence of election auditors to ensure that no secret data is stored, except in the form of printed ballot forms.

### 3.3.3 Teller

The Teller module must be treated as a black box where inputs in an exactly specified format, i.e. batches of encrypted votes, are submitted to it and output in a similarly

3Remember that several Tellers are used in Prêt à Voter and that each must be run by a separate organisation to ensure that a broad spectrum of views are represented.

4This is possible because the RSA onions as implemented always look the same when they arrive at the Teller. Please note that because of time constraints this check has not currently been implemented.

5The system supports dividing the full batch of encrypted votes into smaller batches but this has so far not been used.

6This process is actually started by the Web Bulletin Board collecting random values from the Tellers in such a way that the bitwise XOR of these random values cannot be controlled by any one Teller and is used as the indicator to what links shall be revealed.
Figure 3.2: UML data flow diagram
Chapter 3. An implementation of Prêt à Voter

well-specified format, i.e. batches of encrypted votes, is returned. The reason for this is that ideally several different implementations of the Teller module should be present in a Prêt à Voter system and they should all be run by different organisations. It is important that the Teller does not malfunction and the judged likelihood that in a set of several different implementations of the module, at least one should perform its task correctly, and without leaking any secret information, is high.

The Teller’s main duty is to accept a batch of encrypted ballot forms from the Web Bulletin Board, to remove its layer of encryption from each of these, to shuffle the full batch and then return it to the Web Bulletin Board. The Teller must remember the permutation it has applied to the batch when it shuffled it so that it can subsequently answer audit requests from the Web Bulletin Board to prove that it has not malfunctioned or tried to do so (“cheat”). The Teller should keep track of which encrypted ballot forms it helps to audit during the election phase and not allow these ballot forms to be used in the voting because their secrecy is removed. It should also ensure that it is not somehow tricked into revealing (leaking) any secret data, i.e. the permutation of the ballot forms in the shuffle. Any such secret information that is revealed weakens the secrecy of the election and as the secrecy of the election is based on at least one of the Tellers functioning correctly without leaking any information, it is important that Tellers are run by different organisations and that these implement the module themselves.

3.3.4 Web bulletin board

The Web Bulletin Board can “boot” in any election state and reads and validates this state from the database at startup. This functionality has been put in place to ensure that a Web Bulletin Board server freeze could be rectified simply and quickly with a reboot. The following describes the Web Bulletin Board as it has been implemented.

At the time of start-up the Web Bulletin Board reads in public keys of other system components from its database and it accepts its private key from the operator. It then starts its own server and binds it to the local RMI registry. The web bulletin board now polls the Teller instances specified in the database and if they respond it performs a handshake, meaning that the Teller is validated by the Web Bulletin Board using the public key of the Teller stored in the database. If the handshake is successful the Web Bulletin Board monitors the presence of the Teller and if this connection ever fails it starts trying to reconnect until the connection has been resumed.

---

7Dependent on the number of tellers of course but this number should be adjusted to achieve a desired likelihood — in effect trading complexity (more tellers) for security (a larger number of tellers can leak information before breaking the secrecy of the election).
8This is generally done twice for each batch and each Teller.
9One strategy for shuffling the encrypted ballot forms is actually to sort them by their newly revealed encrypted values.
10Note that it does not necessarily weaken the integrity of the election
11Not that this has been necessary yet.
12These have been specified in the pre-election phase when the diskless workstation was run in the presence of election observers.
13This may seem inefficient but remember that Prêt à Voter 2005 is based on decryption onions using
3.3. The system

In the database is also specified all the Registry, Booth and Audit Machine instances that are able to connect to the Web Bulletin Board. These connect on demand and a connection is not monitored by the Web Bulletin Board, but the credentials of each connecting instance is checked at the start of each connection.

In the Web Bulletin Board GUI is also included options that allow the operator to initiate the process that collects random values from the tellers which is used as the decryption mask in the auditing of the mixing performed by the Tellers. The random values are collected by the Web Bulletin Board as it asks each Teller in turn first for an encrypted commitment to a random value and then for the value itself. The commitments are published by the Web Bulletin Board before it starts asking the Tellers for the cleartext values. These are then published as they are revealed by the Tellers. The Tellers must thus commit to values before they are able to see the values given by the other Tellers, ensuring that they cannot conspire to create a mask which would fail to reveal cheating in the mix by the Tellers. Furthermore, the commitment is signed by each Teller using its private key, making the process auditable. The mask is created by doing an exclusive-or on the bits of all the random values collected by the Web Bulletin Board. As the length of the random values collected by the Web Bulletin Board is lower than the number of links to audit, the mask is repeated so that it includes all the encrypted votes.

The Web Bulletin Board GUI also contains functionality to allow the operator to do a (Single Transferable Vote) tally at the click of a button.

3.3.5 Registry

The Registry module is made up of a simple GUI in which the operator can enter a University Registration Number (URN)\textsuperscript{14}, causing the module to submit a \texttt{voterLookup} request to the Web Bulletin Board. This returns information about the voter (if she is a registered, eligible voter) which is displayed on the screen for the operator to verify. If the operator accepts the eligibility of the voter she asks the voter to extract a ballot form at random from a box in such a way that no-one, including the voter, can see the contents of that, or any other, ballot form. The voter must then fold up the corner of the form to allow the operator to see its serial number. The operator presses the button in the Registry GUI which says “Assign ballot form” and an input box asks for the serial number of the form. Submitting this causes the module to call the Web Bulletin Board (\texttt{assignFormToVoter}), asking it to assign the ballot form to this particular eligible voter. This means that the ballot form can be used to vote and that the ballot form cannot be assigned to any other voter at a subsequent stage. A complete audit trail is kept in the Web Bulletin Board’s database and it will not allow a vote to be entered into its database without it first being assigned by poll station staff to an eligible voter. This does arguably only give any form of election integrity if we trust that the election

\textsuperscript{14}This version of the system was built specifically to handle a University of Surrey Students’ Union election but the search term could be substituted for any other term available in the database of eligible voters.
authority does not maliciously assign forms to voters and submit votes in their names but to combat this threat it is possible in Prêt à Voter to publish the complete list of voters who have voted after the close of the election together with their encrypted votes without this threatening the secrecy of the election\textsuperscript{15}.

If the voter accidentally spoils the ballot form and realises this before it is scanned and submitted, the voter can return to the poll station worker who uses the GUI to cancel the ballot form (\texttt{cancelFormForVoter}) on the Web Bulletin Board\textsuperscript{16}. This adds the cancellation to the audit trail, ensuring that the ballot form cannot be used to vote. This in turn makes it possible to assign another ballot form to the same voter which was previously assigned this ballot form. Depending on the polling station procedures the cancelled ballot form is either shredded, given to the voter to take home or placed in a sealed box and a new form is assigned using the previous procedure.

### 3.3.6 Booth

The Booth module is implemented in Java and and is loaded on a computer in the polling station. When a ballot form has been scanned this data is submitted by the Booth module to the Web Bulletin Board (\texttt{postBallot}) and it receives back a Java object of type \texttt{Receipt}. This contains either an error report which is printed and given to the voter, indicating any reason why the vote could not be submitted or an encrypted receipt which is printed and given to the voter. The encrypted receipt is digitally signed by the Web Bulletin Board and is taken away by the voter for her to use to check the correct inclusion of her vote in the final tally after the end of the election.

If the Booth is able to read some data (specifically the ballot form serial number) but fails in some other part of the process, it performs a cancellation of the form on the Web Bulletin Board (\texttt{cancelForm}) which returns a “cancellation receipt” that the user keeps. The voter is given a new ballot form by the poll station staff when showing the cancellation receipt — and the Registry module allows the voter to be assigned a new ballot form when the old one has been cancelled.

### 3.3.7 Audit Machine

For practical reasons the Audit Machine module is run on the same laptop as the Booth module although it has been developed in such a way that it can be deployed separately from the latter. If a ballot form is scanned without any voter-made marks detected on it, the Audit Machine module is invoked and this makes a request to the Web Bulletin

\textsuperscript{15}This only holds as long as the cryptography does. Please see Section 2.4.7 for a discussion.

\textsuperscript{16}In the literature it is envisaged that the Web Bulletin Board only acts as a method of publishing scanned votes that are to be counted. However, we have opted to also use it as a repository for encrypted ballot form data, specifically the contents of the onions. This means that a reference to this data can be printed on the paper ballot form rather than complete onion as envisaged in the early literature. We also charge the Web Bulletin Board with keeping track of cancelled ballot forms in order to stop cancelled ballot forms from going back into circulation. If such a form was used to cast a vote this would be picked up after the close of the election but in this setup we are able to prevent such mistakes.
3.3. The system

Board (auditForm) for the ballot form to be opened by the Tellers. This causes the Web Bulletin Board to pass the onion(s) of the ballot form to the tellers in order. These perform the decryption of their layers of encryption and return the secret values needed to recreate the onion. All this is stored on the Web Bulletin Board and immediately becomes available on the Website but the Web Bulletin Board returns to the Audit Machine module simply a receipt which recreates the order of candidates on the original ballot form. The voter is able to match the two forms and thus gain proof of the ballot form being correctly formed.

As the ballot form has been audited and all its secret information revealed it cannot be used to cast a vote. Instead the Web Bulletin Board allows another ballot form to be assigned to the voter as the audit trail shows that the original ballot form has been, in effect, cancelled.

The receipt which shows the order of the candidates is signed by the Web Bulletin Board and therefore the values published on the Website can be used to recreate the order of the candidates and this can be compared to the original ballot form (signed by the Administrator module) as well as the audit receipt.

3.3.8 Website

The Prêt à Voter Website module contains functionality\textsuperscript{17} for any interested party to download election information or perform a check on a particular encrypted receipt. The Website works against a non-secret, public database, into which data from the Web Bulletin Board database is exported. No secret information is held in the original database so this division has been made simply to make the Web Bulletin Board more robust to hacking and denial of service attacks.

A voter who has an encrypted receipt can enter the serial number of this ballot form on the Website and see a depiction of the encrypted receipt such as it is stored in the database. The voter can compare the ballot form as it is shown online with the signed encrypted receipt which she holds in her hand. This comparison ensures that the vote which is counted on behalf of that voter after the close of the election is the vote that she cast\textsuperscript{18}.

A voter who has audited a ballot form can bring up the details of this ballot form by typing in its serial number on the Website. This includes the order of the candidates in all the races on the ballot form as well as all the values used to create the onion(s). Anyone with the technical knowledge, or some third-party software, can check that these values do “add up” to the correct ballot form candidate order. The purpose of

\textsuperscript{17}Although not completely functional at the time of writing.
\textsuperscript{18}Strictly the act of comparing these alone does not ensure this: but if the public verification of the system is performed correctly then this check on the input to the mix network ensures that the correct decrypted vote is in the batch of decrypted votes. Furthermore, in order to ensure that no trust has to be placed in the election authority, which runs the Web Bulletin Board, the complete list of all encrypted votes and the mix network should be distributed to several other organisations who publish this data and allows voters to check for the inclusion of their correctly formed vote in the data set that is under the control of this third party organisation.
this exercise is to instill, in the voters, trust that the ballot forms, although all different, are all correctly formed and using them ensures that their votes are not only correctly counted but also verifiable.

### 3.4 Testing and trials

#### 3.4.1 Newcastle trial 2008

A trial was held at Newcastle University in Newcastle-upon-Tyne on the 13th of May 2008. The purpose of the trial was to evaluate an electronic voting system from a usability perspective as part of the joint AROVEv project at Newcastle University, London City University and Toulouse University in France.

**The election**

In order to entice random people to vote in the election and make it worth while to fill out quite lengthy questionnaires, the main race on the ballot form was which of three charitable organisations would get the available pot of money. Two more races were printed on the ballot form and were questions relating to the voter’s confidence in electronic voting systems. The three charities were Barnardo’s, Oxfam and UNICEF. Each charity was invited to send campaigners who would work to get people to vote and to vote for their charity.

**Room layout**

Outside the polling station that had been set up in a very large room close to ground level in a building at Newcastle University were a number (namely three) representatives of the invited charities and they asked passers-by to enter the polling station and take part in the trial so as to secure an amount of money for the charity without giving anything other than a few minutes of their time.

When the voter entered the polling station she immediately encountered a “registration desk” where she was briefed by officials from the trial. They informed the voters that they should first fill in a short pre-voting questionnaire before voting and then verifying their vote (the latter if they wanted to, this was voluntary). Finally they were asked to fill out a longer post-voting questionnaire and, if they felt so inclined, leave their contact details so that officials could get hold of them the following day for a follow-up interview. Note that very few people agreed to take part in such an interview.

**Issues**

A number of voters left the voting booth either without detaching the left hand side candidate list from the right hand side of the ballot form or without destroying the previous. A number of people appeared to have found that they could only safeguard
the secrecy of their votes by taking the candidate list with them as they might not have trusted the available method of destroying the list.

Some people who had not “clicked” that the protected receipt does not leak the contents of the vote because the candidate list is in a random order were keen to put the protected receipt upside down on the scanner so that its content would not be visible to the attendant at the scanning station. We believe it would be beneficial to include in the instructions on how to vote a single sentence that says that the protected receipt resulting from the splitting of the form does not show the contents of the vote.

Technical issues

Using scanners. It is unreasonable to expect people who have never seen the scanning equipment\textsuperscript{19} to immediately be able to use it. Therefore an attendant is required to help with the alignment of the receipt on the scanner, starting the scanning process and so forth. The presence of an attendant and that this attendant is able to see the protected receipt made some people uncomfortable — but interestingly it was very easy to explain to people that the vote is not revealed by the protected receipt and that the attendant could therefore not learn anything from seeing this.

Pre-election testing. In preparation for the election a new ballot form had to be constructed and this is, in this version of the system, a fairly complicated procedure. However, the new ballot form was printed and the optical character recognition (OCR) software was adjusted to be able to read this form. Before going to Newcastle there were a number of ballot forms fed through and the success rate was 100\% both for votes and for audit. The hour before the election further tests were made in the polling station with a 100\% success rate. However, minutes before the opening of the election scan errors started to occur and soon no ballot forms could be read at all. After lengthy search and consulting with other developers, it was determined that two parameters specifying where the OCR software should look for the barcode on the ballot form had not been changed although the barcode had moved a very short distance on the new ballot form. If the ballot form is completely clean in the area between the old barcode position and the new position this would not cause any problems. However, if there was a stray pixel, a tiny smudge or crease in this area the OCR of the barcode would fail. Finding and rectifying this problem unfortunately took two hours after which the election could start.

Lessons

The purpose of the trial was for Human Computer Interaction (HCI) researchers to gather statistical data on the use of an electronic voting system (i.e. Prêt à Voter). Unfortunately there seems to have been some problem with the way this statistical data

\textsuperscript{19}Although the equipment used was off-the-shelf HP printer/scanners that the general public may well recognise.
was collected and it was reported that the data was not useful. However, from a usability perspective a number of lessons could be drawn simply from running the test election:

- Enticing members of the public to cast (honest) votes in a trial election is difficult but offering charities part of a sizeable pot of money in return for their campaigning in the vicinity of the polling station seems to be quite effective in drawing people in. The charities have an opportunity to earn a lot of money for their cause in a few hours work — work quite similar to that which they normally undertake. Voters are able to contribute to a good cause simply by donating ten minutes of their time and do not have to give money directly. At the same time, if there are several different charities in the race, voters feel the election is quite real (as opposed to, say, voting for their favourite cartoon character) since the winning charity will benefit. This sense of “reality” also entices voters to follow protocol, which in Prêt à Voter terms perhaps mainly refers to hiding the ballot form, tearing it and destroying the left hand side.

- The current version of the Prêt à Voter implementation was without any problem able to handle a fairly large number of printer/scanners, decryption and mixing etc on a single laptop, keeping the cost of running a polling station to a minimum.

More lessons are enumerated in Section 3.5.

**Summary**

The trial at Newcastle in 2008, although failing to provide the desired statistical data, was useful and this is a highly recommended way of running non-binding trials of electronic voting systems on a small scale.

### 3.4.2 Demonstrations

After the completion of the first implementation of Prêt à Voter it has been demonstrated at a number of occasions, including:

- Workshop on Trustworthy Elections (WOTE) 2007, Ottawa, Canada

- International Student Electronic Voting Systems Competition (VoComp) 2007, Portland, Oregon, USA

- Frontiers of Electronic Voting 2007, Schloss Dagstuhl, Germany

- Vote-ID 2007, Bochum, Germany

The two main comments received at these occasions were regarding the usability of the system, more precisely the following two aspects:
• The optical character recognition (OCR) of the voter’s mark(s) on the ballot form was dependent on the voter filling out the form using seven segment display type symbols, something that some categories of voters may find hard not only to understand but also to execute. Furthermore, the OCR process was rather fragile and often misinterpreted the voter’s mark(s) (leading to a re-vote) or was unable to interpret them at all.

• The pre-printed ballot form that is filled out, turned into an encrypted receipt and then scanned into the system as a vote is unfortunately very hard to use in systems with large number of races or candidates, i.e. in constituencies with very large ballot forms. Many countries around the world do have very large ballot forms and several politicians have expressed a disappointment with the fact that the hardware requirements would prohibit their use of Prêt à Voter.

More lessons are enumerated in the next section.

3.5 Problems

This section contains descriptions of the problems encountered in making the Prêt à Voter implementation, possible solutions and alternatives.

3.5.1 Single Transferable Vote

Practically, First Past The Post (FPTP) is much easier to scan in and interpret than Single Transferable Vote (STV). We looked at a number of different methods of constructing the ballot form so that it could be reliably scanned in and translated into the correct electronic representation.

One early suggestion was of a grid where the candidates are named in permuted order on the y axis and each position, i.e. first, second, third and so on, were listed on the x axis. The voter would tick one candidate in each of the positions and this would be scanned in. We concluded that this would be too confusing for the voter.

The group also explored different ways of allowing the voter to input her choices through a computer based interface. This would include the ballot form being aligned onto a screen within the voting booth. On the ballot form would be printed the candidate list and a machine readable onion or serial number — simply a bar code. The voter would then be able to on the computer drag and drop the words “first (1st)”, “second (2nd)” and so forth next to each candidate on the paper candidate list. The computer in the booth would then submit to the web bulletin board the ballot form serial number together with the choices made, print a receipt and the voter could shred the candidate list.

This method seemed very promising as it would assist the voter, be accessible and in essence could be the same in the voting booth as it would be online for those voting remotely. However, in this implementation each polling station would require a number of booths, each equipped with a computer, a barcode scanner and a printer. As the
USSU elections were to take place in four polling stations around campus this would require quite a lot of equipment at a fairly high cost. Furthermore, this setup would also require the voters to handle indirection, that is to say the association of each candidate to a symbol (such as a letter or number) and then the marking of this symbol elsewhere, something we were keen to avoid.

We finally made experiments with a paper based ballot form where the ranking was made using a seven segment display, quite similar to that of digital clocks. This method made it possible to have only one computer, one scanner and one printer in each polling station as the voters were able to fill out their forms in secret in booths before scanning them in. The seven segment display would aid Optical Character Recognition (OCR) of the contents of the vote. The experiments went well in that scanning and OCR errors were few and this was therefore selected as the input method.

As a result of this method of accepting the voter’s choices into an electronic form the OCR had to be done at the time of scanning, a receipt must printed with those choices and the voter given the opportunity to check this receipt and reject it if was not the same as the intended vote. This check could be done in public as the candidate list had been removed and it was a matter of checking that the system had correctly interpreted the vote. In our implementation the voter checked the receipt and if it did not match the intended vote, the ballot form was cancelled and the voter issued another form.

3.5.2 Four concurrent polling stations

In order to deal with a number of concurrent polling stations where all eligible voters were able to cast their vote in either one we built a registration module which would connect to the central database and assign a particular ballot form to a particular voter. If this form would be voided or cancelled for any reason it was possible for the poll station worker manning this module to cancel the first and assign another form. The operator used the voter’s University Registration Number (URN) to search for the voter and then typed in a ballot form serial number to assign the form to that voter. The result of this is that when a voter correctly identified herself at a polling station she would be able to cancel any previously cast vote and cast another, making it impossible to vote twice.

3.5.3 Full permutations rather than cyclic shifts

Original Prêt à Voter [76, 20] uses cyclic shifts (although full permutations are possible and fully envisaged) and in Prêt à Voter 2006 [85] this cyclic shift makes it possible to inject the voter’s choice into the onion before it is passed through the re-encryption mix network. As the choice is injected into the onion this means that the onions in a batch cannot be partitioned according to the attached choice. Unfortunately, we have not discovered a way to inject the permutation that represents the voter’s choice in STV into the onion in this way. Instead this choice must pass through the mix network together with the associated onion and thus these may be partitioned according to the attached choices.
If the candidate list is only cyclicly shifted then it is possible to use data on logically consistent votes (for example by looking at the popularity of the candidates in opinion polls) and try to match this information to the choices in a receipt. If a match between the first three candidates for example is made with the voter’s “1”, “2” and “3”, then it can be determined with high probability that the vote is for those candidates in that order. This matching is possible because the order and distance between candidates in the shifted candidate list does not change with the cyclic shifts and therefore the voter’s marks might only correspond to a single, logically consistent vote.

To combat this we were not able to use cyclic shifts but had to use full permutations. This in turn meant that we could not use the re-encryption mix network of Prêt à Voter 2006 but had to revert to the decryption mix network of original Prêt à Voter. The latter suffers both from the chain voting and the administrator knowledge problems [90].

### 3.5.4 Paper trail

In Prêt à Voter as defined by the literature the voter is allowed to leave with the right hand half of the ballot form as an encrypted receipt. When this half had to be left in a ballot box this meant we were required to print another receipt for the voter to take away.

The process thus became as follows:

1. The voter is assigned a ballot form
2. The voter fills out the ballot form in secret and detaches the two halves
3. The encrypted receipt is scanned
4. A receipt is printed
5. The voter checks that this receipt is correct
6. If it is correct then the voter staples the two halves back together and these are put in the ballot box\(^\text{20}\)

### 3.5.5 Allowing the vote to be spoilt

Something that appears to have been overlooked by many constructors of electronic voting systems is the (arguably) democratic right to spoil a vote. In order not to comply when being subject to the chain voting attack for example, the voter’s last resort is to void the ballot by making some mark on it or filling it out erroneously. In some places around the world voters take this opportunity to perhaps “send a message of discontent” rather than casting a vote. Furthermore, in countries where voting in

\(^{20}\)Please note that this step is necessitated by the USSU requirement that a human readable audit trail exist against which the electronic voting system can be checked — it is a serious threat to election secrecy and should therefore be removed as soon as confidence in the system has been established.
elections is compulsory (such as Australia) voters may take the “right” to spoil their vote without anyone knowing quite seriously.

However, in our implementation we used OCR and had to make sure that what the machine interpreted is what the voter intended. This OCR thus had to take place immediately after the scanning of the ballot form and a receipt printed which the voter was able to check against her own ballot form half. This meant that the OCR software had to communicate to the voter that it had received something that did not make sense. Also, it would disregard marks outside the voting grid itself that might void the ballot form in other circumstances and only interpret the numbers within the grid\textsuperscript{21}.

\subsection{3.5.6 More than one race on a ballot paper}

The USSU election for which the implementation would first be put in use had seven races: president, four vice-presidents and two referenda. Traditionally, and for the sake of simplicity, all current races have always been printed on the same ballot form. In Prêt à Voter the form consists of a candidate list to the left and a grid for the voter to insert the choice to the right. In the implementation we had to have seven such forms on a single piece of paper. Because of the small number of candidates in each race this was in fact possible.

In order to cope with this we had to add one level of abstraction: that, which popularly is a ballot form, became a ballot form paper, consisting of seven ballot forms. Each of these ballot forms would have its associated onion and so forth, but be referenced by a single serial number, that of the ballot form paper. When a voter was assigned a ballot form by a clerk this in fact meant assigning a ballot form paper. When the polling station software would submit the vote to the web bulletin board it would do so with one serial number (that of the paper) and seven permutations representing one vote in each race.

We devised the system such that when a vote was submitted in this fashion each permutation would be assigned to its ballot form in the database. In the decryption and tallying phases no reference would again be made to which paper a ballot form might have been printed on, which meant that the simplicity of original Prêt à Voter was regained in these phases.

\subsection{3.5.7 Partition according to number of choices}

Another way that the votes may be partitioned as they go through the decryption mix network is by the number of choices the voter has made. To comply with the general principle of STV the voter must be able to rank any subset of all the candidates available. Some voters may prefer to give support to only one candidate while others may rank all. The batches of onion rank pairs can be partitioned according to the number of choices

\textsuperscript{21}In order to allow a voter to spoil the vote we discussed the possibility of inserting a \textit{STOP TRANSFERRING} candidate in each race, which, whenever given the first rank by a voter, would lead to the vote not being tallied. This would also be a solution to the partitioning of the votes according to the number of choices made by the voter, see Section 3.5.7.
made. Any such information leaks have a negative impact on ballot secrecy and should be avoided.

The solution that we discussed but because of time constraints did not implement was to include a *STOP TRANSFERRING* candidate (just as in Section 3.5.5) in the list. The voter would then make her choices and when she is finished continue ranking all the other candidates randomly, but starting with the *STOP TRANSFERRING* candidate. Alternatively this can be done automatically.

If the rest of the candidates are ranked randomly (but no votes are transferred to them because the *STOP TRANSFERRING* candidate appears before them) then the receipt will not be exactly the same as the voter filled it in. Although it is possible to explain why this is, this certainly makes it less straightforward for the voter to see why the receipt encrypts her vote.

### 3.6 Summary

Developing the first implementation of Prêt à Voter has been an amazing journey: from a series of theoretical papers to a first version system which has proven itself to be quite usable. The gap between theory and practice is immense but bridging it satisfying.

In our work with this version of the Prêt à Voter implementation we have focused on the issues relating to implementing STV elections in electronic voting systems and our contribution here is to the understanding of this problem. Other practical issues that we have looked into is the problem of having a large number of races and candidates in a paper-based system (this is explored further in Chapter 5).

### 3.7 Contribution

This chapter is a case study into the implementation of an end-to-end verifiable electronic voting system. We found that because of the Italian attack, implementing Single Transferable Vote in a system where the secrecy of the vote is based on a cyclic shift, such as Prêt à Voter, is impossible without decreasing the coercion resistance of the system significantly. The case study also focuses on practical aspects of implementing a verifiable voting system in the form of often difficult and unforeseen requirements from the organisation holding the election. In this case this was in the form of the requirement to keep the original ballot forms as a paper audit trail and the requirement to run four concurrent polling stations where voters are allowed to choose any one.

We also gave some detail in this chapter on our exploration of the possibility of intentionally spoiling a ballot form, which is important for the system’s coercion resistance as well as to those voters who do not wish to cast a vote but who are required to do so by law.
3.8 Future directions

The next version of the system must aim at implementing Prêt à Voter 2006, the re-encryption mix network version. This setup is more advanced in the theoretical security and accuracy features that it provides but uses cryptography which is more complicated to implement in the way envisaged in the theory. There is no doubt, however, that the next version of the system will not only be very usable but will also be based on the most recent results in end-to-end verifiable electronic voting systems research.
4 Component based electronic voting systems

4.1 Introduction

It appears that each researcher in the field of Electronic Voting Systems contributes to some particular aspect but re-builds the whole system when they wish to implement this rather specific contribution. The idea presented in this chapter is that in order to build an e-voting system we simply add certain distinct pieces together — and in order to improve on a particular system we swap one distinct piece for another that fits into the same slot. In short, we are proposing that we start thinking about electronic voting systems as being component based.

A benefit of this thinking is that for each component slot, i.e. a place in a layer where a component of the system can be slotted in, it is possible to define the method of assessing the computational complexity of that component as well as performing a threat analysis. Similarly, if we agree that all components of an electronic voting system should be verifiable and/or auditable then it is possible in this configuration to define for each component a method of verification or audit. When an author then makes changes to one or more components it is possible to in effect “re-run” checks on those components or to employ the same verification method on a different component.

4.1.1 Domains captured

As the reader goes through the description of the different layers, she may realise that the components in those layers are not strictly components from a computer systems design perspective. Instead, the components capture the full spectra of the design and implementation of an electronic voting system — in other words they capture the following domains:

- Computer system domain
- Human (user/voter) domain
- Legislative domain

We do not make distinctions between components in the hierarchy that we propose, that is to say that we do not treat components from different domains differently. To a developer of electronic voting schemes this will be perfectly natural as her overview of the system is total. When she considers all the components then they must all fall
into place and completely make up the (correct) system. In fact, it is the duty of the developer to ensure that a component that is put forward does fulfill all the requirements on that component.

Others may look differently on the component based electronic voting system. A programmer (implementer) may look only at those components that are implementable in software. Similarly someone who might be consulted on the legal implications on the configuration of an electronic voting system may only consider components from the legislative domain. Both these non-technical developers are examples of people who must be able to trust that the system as a whole depends on each of their respective components and that the configuration of their components are reflections of the requirements and conform to the restrictions of those same components.

4.1.2 Cascading changes

When a system is considered to be component based we believe that changes can easily cascade down the different layers, all the way down to the implementation. In a trivial sense this might simply be that when the developer changes the structure of a component this automatically becomes a list of changes for the implementer to make in the actual code. However, it is easy to see that changes made to one component will only result in changes to the implementation of that same component and not to surrounding or distant components, restricting the work needed to implement the changes.

4.1.3 Discussion on the reach of the proposed methodology

The scheme proposed here is an extension of that published as [52], which has been a foundation to schemes such as that proposed by Popoveniuc and Vora [70]. This scheme is offered as is and it is conceded that it is not in a complete form. The ideas presented here (such as the proposed components and the domains into which they fall and that the model captures, the relationships between those components and the types of relationships components may have) are in some cases not fully explored and require further research before being fully functional.

For example, the types of relationships shown in Figure 4.1 have proven hard to use in the examples given in this chapter and may have to be reconsidered (such as has been done in [70]). This is why they are not used in the example — but the examples are still given to help show the foundation that this chapter hopes to give to component based electronic voting systems. Similarly, the example decomposition of the Punchscan system in Section 4.4.1 is given as a precursor to a more in-depth analysis that may become possible in the future.

4.2 Component hierarchy

We suggest that an electronic voting system is made up of parts from four comparatively separable and distinguishable layers, each of which builds on the services provided by a
4.2. Component hierarchy

Table 4.1: Layers and components

<table>
<thead>
<tr>
<th>Layer</th>
<th>Components</th>
</tr>
</thead>
<tbody>
<tr>
<td>Human layer</td>
<td>Voter registration</td>
</tr>
<tr>
<td></td>
<td>Ballot form configuration</td>
</tr>
<tr>
<td></td>
<td>Polling station layout and management</td>
</tr>
<tr>
<td></td>
<td>Verifiability front-end</td>
</tr>
<tr>
<td>Election layer</td>
<td>Election method</td>
</tr>
<tr>
<td></td>
<td>Election management system</td>
</tr>
<tr>
<td></td>
<td>Voter-ballot box communication channel</td>
</tr>
<tr>
<td>Computational layer</td>
<td>Cryptography scheme</td>
</tr>
<tr>
<td></td>
<td>Anonymisation strategy</td>
</tr>
<tr>
<td></td>
<td>Tallying procedure</td>
</tr>
<tr>
<td>Physical layer</td>
<td>Hardware authentication method</td>
</tr>
<tr>
<td></td>
<td>Publishing strategy</td>
</tr>
<tr>
<td></td>
<td>Transfer method</td>
</tr>
</tbody>
</table>

lower level layer. We propose to name these layers of the model as follows:

1. Human layer
2. Election layer
3. Computational layer
4. Physical layer

The physical layer enables the reading in of voter choices and the transfer and publishing of the same. The computational layer contains that which is chiefly encapsulated in software and which relies on the physical services of the lower level. The election level encapsulates all those options and configurations relating to the election system being implemented by the two lower levels. Finally the human layer deals with those aspects of the electronic voting system which face the voters.

The layers and components are shown in Table 4.1. We will now go through these layers from the bottom up so as to first provide a solid foundation and then explain how all the aspects of electronic voting systems may fit into this model.

4.2.1 Physical layer

The physical layer supports all other layers with a physical (or semi-physical, see below) infrastructure to facilitate different hardware based aspects of the system. We divide the physical layer into the following components:
Hardware authentication method

The different physical components of the electronic voting system must, in some cases, be identified using some authentication strategy, for example an asymmetric key pair and a public key infrastructure (PKI) with established trust among voters.

Publishing strategy

Voter verifiable electronic voting systems depend on the information that voters need to perform the verification being delivered to them in a way that they can trust. An observation is that a single source may be an unreliable publishing strategy but a combination of web sites, newspapers and other independent outlets may be more reliable.

Transfer method

All electronic voting systems capture some information from the voters and transfer it to some repository, be it central or distributed, before the information is interpreted and tallied. In this slot fits methods for transporting such digital information from polling stations to such repositories, for example Secure Socket Layer (SSL)\(^1\) over the Internet or storage on flash drives that are manually distributed.

4.2.2 Computational layer

In the computational layer are defined all the components that are performed explicitly by software. These include the following components:

Cryptography scheme

The cryptography scheme employed probably underpins much of the operation of other components in this layer so there is an intra-layer dependence. However, the cryptography scheme is more easily defined in its own component after which the other components of this layer (and others that may rely on it, although restricting the use of the cryptography scheme to the computational layer only would greatly simplify the definition of a scheme in this model) may refer to it. One example of this may be where the cryptography scheme employed is Elgamal and re-encryption mix networks are used as anonymisation strategy in that component.

Anonymisation strategy

Electronic voting schemes that are both “receipt free” and voter verifiable must use some anonymisation strategy to break the link between an encrypted receipt and the plain

---

\(^1\)By choosing to consider SSL part of the physical layer in our model, we are able to reduce the complexity of higher layers. It is acknowledged that other models may not place SSL in the physical layer.
4.2. Component hierarchy

text vote\textsuperscript{2}. This strategy is captured in this component although it most likely is heavily dependent on the cryptography scheme defined in the previous. However, in some cases the anonymisation scheme may rely on “any” cryptographic scheme and so changing the cryptography scheme component does not necessarily interfere with the anonymisation strategy component. One example of where this is true is where the anonymisation strategy is a re-encryption mix network. It is logical that the cryptography scheme may be Elgamal or Paillier and one scheme may be removed and the other slotted in without this requiring the anonymisation strategy to change.

**Tallying procedure**

The tallying procedure component is simply an encapsulation of the procedure and software that performs the tallying of the decrypted votes. This may seem like a trivial component at first glance but when the component properties that we introduce below are considered it appears that a verification strategy and computational complexity analysis for example is done better across this component than across some external “election administration” software\textsuperscript{3}.

### 4.2.3 Election layer

This layer is very much derived from the laws that govern the election. It seems likely that when this layer is implemented it may result in some components not being turned into code and others may be external software. As an example of the latter we can take that the implementation of a component may actually be a formal specification of a piece of software that can successfully audit an election. This is then published and any number of interested parties may write their own piece of software.

The election layer consists of the following components:

**Election method**

This component contains a specification of the election method that is used. If the model is scrutinised as a whole it may be beneficial to have in one part of it the specification of the election at such a high level.

**Election management system**

All schemes require an election to be set up by some clerks or civil servants — or politicians or some trusted third party. This component is, much like the previous,  

\textsuperscript{2}Schemes that use the homomorphic property of an underlying cryptographic scheme to tally without doing decryption simply breaks the link between encrypted votes and tally without presenting decrypted votes.

\textsuperscript{3}We note here that it is hard, if not impossible, to place cryptography in its own component — in this case the tallying algorithm may, in some systems that do not first decrypt votes into plain text, be highly dependent on the cryptography scheme used.
incorporated into the model for completeness although it does require to communicate with other components.

**Voter-ballot box communication channel**

This may seem like a much more low-level component but in fact it is included in this layer to enable high-level definition of the secrecy of the election. In other words it is possible to describe here the full procedure that the voter goes through to cast a vote in a way that is understood not only by the developers of the system but also the general public. This definition may then be used as reference when other components at lower levels in the model are composed.

**4.2.4 Human layer**

Although some components in the previous layer have become human readable and in fact only serve to further the understanding of the lower layers (or, arguably, define the lower layers in a human readable way, which then cascades down) this layer deals with all those aspects of the electronic voting system that are facing the voter. Therefore the components we expect to find in this layer are:

**Voter registration**

The procedure for and timing of voter registration along with the criteria that makes a voter eligible to register are all enshrined in the law relevant to the use of the implementation of the modelled system. They are included in the model in this component so as to provide completeness. As mentioned in Section 4.1 the completeness of the system is entrusted to the electronic voting system developer and so a change in this component must trigger any necessary changes to other components — and these may of course be components that are eventually implemented in code.

**Ballot form configuration**

The ballot form configuration is very important in any electronic voting system and it seems likely that this component has links to other components, for example the cryptography component in the computational layer.

**Polling station layout and management**

This is another component that is directly derived from the applicable national and regional legislation. If the component is precise, by which we mean it is directly derived from legislation which sets out exactly the layout and management, then this may dictate the configuration of other components in the model. If it on the other hand is permissive then this would imply that only a few guidelines are given and that if necessary the layout and management of polling stations may be adapted to suit the electronic voting system.
4.3 Component properties

Verifiability front-end

To present a unified verifiability front-end to voters and concerned groups, this component acts to capture the requirements of such a front-end at a high level and thus restrict other components that must comply therewith.

4.2.5 Component interaction

As has now become abundantly clear it is impossible to make all components of an electronic voting system fully autonomous — and we simply are not striving to do that. Instead, by making each component as distinct and autonomous as possible and then providing links between components we can reach some compromise between all the good properties brought by a component based model and the necessity of component interaction.

The links we propose are not communication links as such, in fact we regard the communication between different implemented components to be modelled at a lower level. This is supported by the fact that only some components at this level of abstraction are implemented and only some of those communicate — in fact some of these components become entangled in an implementation, for example the cryptography component which is used in an implementation of the anonymisation strategy component in the computational layer.

The component links that we propose are of only two kinds and each link is a unidirectional vector. The link carries some fact attached to it and it either defines a restriction by one component on another or a permission given by one component to another.

The links are not very complex and can easily be illustrated in a graph, an example of which is shown in Figure 4.1. If the underlying fact or facts are coded in some way (perhaps just a unique numerical reference) within the imposing component then the link can easily be shown in a graph by using an arrow with a name corresponding to that code (reference).

4.3 Component properties

The components in this model of electronic voting schemes have a number of properties that can be defined much easier than similar properties for the system as a whole. The properties that distinguish one component from another include:

4.3.1 Layer and slot into which it fits

The hierarchy of components is determined before the model is set up for a particular electronic voting system, and logically declaring the different parts of “a” system before filling in those slots with a particular implementation does seem to restrict the developer in some sense but on the other hand it does result in a system that can be more easily explained and developed. From a verifiability perspective such a system could also have
4.3.2 Origin, requirements and constraints posed by the slot

This set of properties describe the requirements on the component which fits into the slot. As the requirements come from the model, rather than being made up on the fly by the developer of the component, it is possible to change between two components.

4.3.3 Verification strategy

Because of high verifiability demands in electronic voting systems (see Section 2.2.3) the requirement on every single component of the component based electronic voting system is that there exists some verification strategy.

This may be one of the most attractive aspects of the component based methodology. By providing some overview of the verifiability of each component the developer can show that the full system is verifiable. By slotting one component out and another in, the same verification strategy can still be in place, reducing the work needed to compose a new component for a particular slot.

As an example of this we can mention the anonymisation strategy component in the computational layer. Let us say that a decryption mix network makes up this strategy.
(using the cryptography defined in the cryptography scheme component). The verification strategy on this component is a variant of zero knowledge proof. If a developer slots this anonymisation strategy component out and replaces it with a re-encryption mix network instead, the verification strategy remains the same.

The requirement to have a verification strategy for each component is not restricted to components that are implemented in code but is applicable also to components in the election and human layers. For example, a component in the human layer may have a verification strategy which mandates a group of auditors to check the work performed in a procedure described by the component.

### 4.3.4 Location in the system (authority-close or voter-close)

A component has a particular position in the system as a whole and this may be described by whether it is close to the voter, and controlled by her, or if it is close to an election authority and controlled by them.

One example of this might be the ballot form component in different versions of Prêt à Voter. In Prêt à Voter 2005 [20] the ballot form is quite authority-close in that it is set up by an election authority before the election and this authority holds all information needed to decrypt the receipt. To combat kleptographic attacks [82] Prêt à Voter 2006 [85] introduces re-encryption mixes whereby the creation of the decryption information is distributed among a number of trusted parties. The form is printed on demand by the voter in the booth and can thus be regarded to be voter-close.

### 4.3.5 Threat analysis

Each component, just as it has a verification strategy, can have a threat analysis attached to it. This analysis holds for any component that is slotted into the same place. The output of such a threat analysis is a number of requirements on the component.

### 4.3.6 Computational complexity analysis

Seemingly more dependent on the particular component, the computational complexity analysis property of the component is part of the computational complexity analysis of the complete system. By decomposing the system into smaller parts this analysis also becomes much easier, aiding the implementation onto hardware. For example, consider that an encrypted vote in a particular system may consist of a number of layers of encryption and that each encryption and subsequent decryption of such a layer takes a certain amount of processor time. The difference between two different encryption schemes, in regard to the processor time needed, may be substantial and this is then magnified by the number of votes cast in the election.
4.4 Examples

In this section we apply this model to two already existing schemes, in the first instance Punchscan [71, 28, 68] and in the second Prêt à Voter [20, 76, 82, 85]. These decompositions are included to illustrate that the model would be applicable to schemes that already exist and that the further development of those schemes could be done in a component based methodology.

4.4.1 Punchscan: an example decomposition

Punchscan relies heavily on a central election authority to set up, manage and guarantee the safety, security and reliability of the voting system. The anonymisation strategy can be audited using a zero knowledge proof method.

**Human layer**

**Voter registration**  The voter returns a form delivered to her house by local government. When she is registered a polling card is delivered by post.

**Ballot form configuration**  The ballot form is made up of two pages, the first has holes through which the second can be seen. On the first page is printed the races and the candidates in those races. Next to each candidate can be found a symbol which matches a symbol on the second page, visible through one of the holes. To vote the voter marks both pages using a bingo marker over the second page symbol which corresponds to the candidate she wishes to vote for.

**Polling station layout and management**  When the voter enters the polling station her name is checked against the register. She identifies herself using the polling card.

**Verifiability front-end**  Voters can check their receipts on a web site as well as in the local media. The audit of the full election can be viewed online or through media reports.

**Election layer**

**Election method**  The local representative is elected by first past the post and thus each voter gives her vote to a single candidate.

**Election management system**  Before election day the election authority which oversees the election calls a meeting of election officials and representatives of the different political parties. In this meeting the election is set up on a diskless workstation running trusted, audited software. The output of this process is a database which is kept secret by the election authority for the purpose of decrypting votes after the election together with instructions to a print company which will print the ballot forms.
4.4. Examples

**Voter-ballot box communication channel**  The voter is allowed to fill out the ballot form in the privacy of a voting booth. Before leaving the booth she is able to create the encrypted receipt, making it hard for anyone to capture information from both layers of the ballot form. The encrypted receipt is scanned (with the aid of poll station workers) and transmitted electronically.

**Computational layer**

**Cryptography scheme**  Although no cryptography as such is used in Punchscan, the vote is hidden behind a number of random translations that are in turn captured in a number of interlinked decryption tables. There exists one translation for each of the two pages in each decryption table (the translation may be the identity mapping) and a number of decryption tables may be linked together.

**Anonymisation strategy**  The decryption tables are set up in advance such that when a ballot form is made into an encrypted receipt, the application of all the translations for that form in all decryption tables reveals the voter’s intention. In order to anonymise a plaintext vote the full batch of encrypted receipts are secretly mixed after the application of each decryption table. The decryption is audited by making the authority commit to the decryption of a batch and then challenge it to reveal a number of (but not all) links between the input and output batches. By not auditing all links the full link from an encrypted receipt through to a plaintext vote is broken at some stage.

**Tallying procedure**  The encrypted receipts are published to the web, the election authority applies its decryption tables and anonymisation strategy and publishes the result of each step. When the plaintext votes appear at the end the authority, and anyone wishing to check the result, can perform the tally.

**Physical layer**

**Hardware authentication method**  Each polling station machine is issued a cryptographic key pair and an identity.

**Publishing strategy**  All encrypted receipts received by the central repository are published, in some predetermined batch mode, to a publicly accessible read-only online resource.

**Transfer method**  Communication between the polling station and the central repository is encrypted using SSL.

**Component relationships**

As described above components can either pose restrictions on other components or, in order to explicitly specify the space in which developers have to manoeuvre, can grant
other components permissions. We continue this example by showing these relationships between the components of Punchscan above.

As we are aware that the anonymisation strategy is heavily influenced by the election method, i.e. it may be simpler to implement an anonymisation strategy for a first past the post election than STV, we make it explicit that the election method is first past the post:

<table>
<thead>
<tr>
<th>Permission</th>
<th>Election method</th>
<th>to</th>
<th>Anonymisation strategy</th>
</tr>
</thead>
<tbody>
<tr>
<td>First past the post. The election method is first past the post and no other election method is used in this constituency.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The end-to-end verifiability of Punchscan is based on a number of items of data being created and committed to before the election. Therefore the publication strategy component must allow for this publication⁴:

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Anonymisation strategy</th>
<th>to</th>
<th>Publishing strategy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pre-committed values. Election data will be created in a series of meetings of election officials before the start of the election and this data will be committed to by these officials. The publishing strategy must be able to publish a set of these commitments.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

No cryptography is used but the verifiability of the anonymisation strategy is heavily dependent on a hashing algorithm:

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Anonymisation strategy</th>
<th>to</th>
<th>Cryptography scheme</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hashing algorithm. A well-known, secure message digest (hashing) algorithm must be available.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The relationship between these components are shown in Diagram 4.2. Please note that the relationships between the components may be different for decompositions of other systems or using different components. Working with the component based model, the developer uses both existing relationships and the changes in relationships to correctly integrate a new component.

4.4.2 Changes to cryptography component in Prêt à Voter

In the original Prêt à Voter [20] the cryptography scheme component is RSA, providing services to the anonymisation strategy component, which consisted of a decryption mix network. This system suffered from the authority knowledge (or chain voting) problem [82] and thus the next version [85] slotted out the RSA cryptography component and slotted in an Elgamal cryptography component⁵. This new component was able to support a re-encryption mix network in the anonymisation strategy component and this in turn meant the the ballot form could be printed on demand in the booth, affecting the ballot form component.

⁴This is quite similar to a use case in traditional modelling of information technology systems.

⁵Substantial other changes were also made to the scheme at that time.
4.4.3 Using Punchscan style ballot form in Prêt à Voter

Quite interestingly we can show in this section that the “traditional” Prêt à Voter [20] ballot form is completely interchangeable with the “traditional” Punchscan [68] ballot form (this has also been shown by [92]). The ballot form is thus a prime candidate for a component in the electronic voting system decomposition.

The ballot form in Prêt à Voter, shown in Figure 4.3, has a randomly ordered candidate list in the left of two columns. The right column consists of a grid where the voter marks her choice(s). Below this grid is printed the onion which encapsulates the order of the candidate list under a number of cryptographic layers. When the form is torn along a vertical perforation between the two columns and the left column shredded, the encrypted receipt remains. The onion value uniquely identifies the ballot form.

The Punchscan ballot form consists of two pages, shown in Figure 4.4, on the first of which is printed the candidate list in the canonical order. Next to each candidate on this page is also printed a symbol which corresponds to the same symbol shown, through holes in the first page, on the second page. The voter marks her choice using a bingo marker. This makes a mark on both pages at the same time and when one page is randomly selected and the other shredded, what remains is an encrypted receipt. The ballot form serial number printed in the top-right corner uniquely identifies the ballot form.

In both schemes the encrypted receipt is scanned and transmitted digitally. We can now describe both these forms as a component with the following configuration:

**Layer and slot into which it fits**

Human layer, Ballot form configuration slot.
Chapter 4. Component based electronic voting systems

Figure 4.3: The Prêt à Voter ballot form

Figure 4.4: The Punchscan ballot form
Origin, requirements and constraints posed by the slot

The ballot form must list the candidates in the race for which it is valid on one half of the form. The other half must accept the voter’s intention. If the two halves are separated one remaining half must not reveal the candidate(s) for which the vote was cast but must be decryptable to reveal this information.

The form must be printed on paper, security paper is permitted. It must be scannable and shreddable.

Verification strategy

A reference printed on the form must uniquely identify it so that the voter may search for it in an online resource after the close of the election. During the election the voter may also use this reference to audit a form which will not be used for voting.

Location in the system (authority-close or voter-close)

The ballot form is created in advance by an election authority, distributed under guard to safeguard the secret of the form and picked out at random by the voter.

Threat analysis

The ballot form must never be seen by anyone other than the voter before one half is removed to form the encrypted receipt as this may remove the system’s coercion resistance [82]. The ballot form is therefore to be distributed within an envelope that some physical procedure must guarantee that only the voter may open within the booth.

Computational complexity analysis

Creating the ballot form involves creating some decryption information, applying it a number of times over and then printing the form.

4.5 Discussion

This section provides a quick overview of some of the caveats with the component based model that we have foreseen.

4.5.1 Impossible to have strict boundaries between components

Defining electronic voting schemes as component based systems provides researchers with the opportunity to focus development on a particular component or to compare two different components that fit the same slot to determine which is best. However, when the component based model is turned into an implementation there may be complications. If an implementation is made of one particular component based model and one or more components are changed in the model then cascading these changes to the
implementation may not be trivial. For example, the implementation of the mixing strategy may be heavily dependent on the cryptography scheme used and a change of the latter most likely results in the change of code in the earlier.

### 4.5.2 Restrictions on the developer

Placing restrictions on the developer brings framework and structure and it also enables the re-use of components (both as specifications and as code) and the extension of schemes by subsequent developers.

### 4.5.3 Requirements must come from the model

A developer of a component may look at the technical contribution of that component and make up the requirements of the component from that. This is easy to do but then suddenly the resulting system does not contain components that may be re-used or changed easily. Therefore it is important that the electronic voting system developer looks at the system as a whole and fully defines the requirements on each component before proceeding to create those components.

### 4.6 Summary

We have presented a component based methodology for developing electronic voting systems which aims to model all aspects of the system, not only technical. By not changing the complete system we hope that developers may be encouraged to look in depth on a particular component or set of components, providing a complete threat analysis as well as verification strategy for each. The model presented here consists of four layers in which components ranging from the very technical underpinning the communication of various parts of the system to the very non-technical collating requirements from legislation are situated.

### 4.7 Contribution and future directions

The contribution of the work presented in this chapter is to introduce a hierarchy of components, organised in layers, that is able to hold information about all aspects of an electronically tallied election. The intention has therefore been to allow the structured collection of all the information necessary to run an election, using an electronic voting system, in a particular country. Such a repository of information, although a model, is not readily translated into code (i.e. the implementation of an electronic voting system) and the intention of the model presented here therefore cannot be to replace more traditional component modelling of those parts of the system that are in fact implemented in code. Instead it is acknowledged that such modelling should be done of a subset of the components introduced here. The justification for the approach taken here is that the modelling, in this way, of even non-technical aspects of running the
4.7. Contribution and future directions

electronic election will make it more straightforward to build models of (implementable) systems that meet the requirements set out in law.

Another approach to a component based view on electronic voting systems is introduced by Popoveniuc and Vora [70] when they show that Prêt à Voter, Punchscan, the Chaum visual cryptography based system [17] and Scantegrity II [19] are composed of front-ends and back-ends with similar properties where components can be interchanged between the system (as in the example above).
5 Simple and secure electronic voting with Prêt à Voter

Implementing Prêt à Voter as it is described in a series of papers [76, 20, 82, 83, 85, 80, 55, 56, 108, 54] has an associated set of fairly hard problems that were not within the scope of these papers, such as reliable optical character recognition (OCR), multi-page ballot forms in elections where there are many candidates contending many different races, chain of custody issues relating to pre-printed ballot forms, key distribution problems relating to on-demand printed ballot forms and so forth.

Our impression is that politicians and civil servants, in Europe and probably around the world, are concerned with the accessibility and applicability of electronic voting systems to a higher degree and cutting-edge security technology to a lesser degree than seemingly realised by researchers in the electronic voting field. Consider for example the impossibility for a civil servant in a country in continental Europe where there may, for example, be 28 candidates in each of seven races contended on the same ballot form to implement Prêt à Voter 2005 or 2006 — the ballot form is simply too large to be scanned.

Further anecdotal evidence suggests that a major contributor to decisions to use electronic voting in Europe is to simplify the process. For example when the City of Hamburg, Germany, changed its electoral law it almost became a necessity to use some form of electronic counting of the votes as this would take days and weeks to do by hand [94]. The decision was taken to implement a completely new system based on Anoto pens1 and although this system was very accessible and had some procedures to safeguard the accuracy of the election, it seems it lacked sufficient technical guarantees of this.

We suggest that when implementing a real-world electronic voting system it is necessary to, to some degree, use the age old accessibility measurement “would my grandmother understand how to vote”. Please note that we do not suggest that the grandmother in question necessarily understands what technology, procedures and cryptography propagate and secure the vote, but we do suggest that she understands how to cast her vote and that she be not unnecessarily alarmed by some feature of the electronic voting system and as a consequence starts to doubt the secrecy of her vote or the accuracy of the tally.

This Chapter proposes a configuration of the Prêt à Voter electronic voting system in its later guises with emphasis on usability, accessibility and simplicity. In practical terms this means giving the voter help forming her encrypted vote when there are many,

1This type of pen writes as a normal pen but when used on a paper with a special, invisible, reference grid, a camera at the tip of the pen registers the strokes made by the pen and digitises these.
many candidates and races by allowing her to safely use a computer to form the vote.

5.1 Introduction

In this section we describe the properties of end-to-end verifiable systems and introduce the procedure/technology concept.

5.1.1 End-to-end verifiability

The will to elect leaders and representatives stem from a mass of people, equal, who have organised and created states and institutions to serve the population. From this philosophical point of view some may say that once leaders were first democratically elected, they created election authorities and thus these are trustworthy and able to run fair elections for the people. Others are more reluctant to place such trust with such authorities. Consider for example some of those states in the world today that wish to disguise an undemocratic rule by holding general elections that are unfair. The most effective weapon against this is the disposal of the world’s truly democratic nations is election observation.

However, election observation is a very blunt instrument with tremendous organisational and budgetary requirements. Although essential, election observation can only function as an audit of the procedures in place to safeguard the election and it is impossible to know, or prove, that the audit is sufficiently complete to allow conclusions to be drawn about the secrecy and fairness of the election.

This suggests that it would be more beneficial, if it was possible, to audit the election as a whole rather than some subset of the procedures involved. The ability to audit the whole election and (perhaps mathematically) prove that the outcome is exactly as indicated by the voters on election day has been given the name end-to-end verifiability and there exist many systems aiming to do this [4, 5, 7, 8, 10, 17, 20, 25, 28, 30, 46, 51, 63, 64, 71, 73]. There may be other ways of achieving this but we consider end-to-end verifiability a combination of two other: voter verifiability and public verifiability.

**Voter verifiability** The voter is given a receipt which she can use to check after the close of the election that her vote has been included in the tally. In order for the system to be coercion resistant the receipt must not reveal the vote.

**Public verifiability** Any interested person or organisation can, perhaps using software, check that all the encrypted receipts are properly decrypted into plain text votes and that these are tallied correctly.
5.1.2 The procedure/technology mix

We confess that we would rather employ a technological solution to security issues in electronic voting systems than a procedural one\(^2\), but here feel obliged to introduce the \textit{procedure/technology mix}. This is simply the mix of technology, procedures and people that constitutes any electronic voting system.

In the previous section we claimed that the use of end-to-end verifiability would render the auditing of procedures and people obsolete. This is certainly true regarding the correctness of the outcome of the election, it is simply possible to prove whether the reported outcome is correct or not and if not, find the source of the error.

However, the \textit{secrecy} of the election is, of course, a kind of property that once leaked cannot be “proven” back to secrecy. Furthermore, end-to-end verifiability is unfortunately very hard to achieve with technology only. Consider for example a theoretical system, the accuracy and secrecy of which depends on each voting device having its own secret key. The distribution of these keys are in fact a procedural solution to both the accuracy and secrecy problems!

Furthermore, the registration of eligible voters and the identification of voters in the polling places are, in many places, heavily based on procedures (identity cards are, for example, checked by poll station staff and compared to the list of eligible voters. It therefore seems logical that the secrecy of the election is safeguarded by some mix of technology and procedures and we advocate a use of procedures to increase the accessibility of the system where a technological solution would reduce it.

5.1.3 Definitions

\textbf{Web bulletin board} To facilitate the verifiability of the system a \textit{web bulletin board} is used to publish data at different steps throughout the election process. The web bulletin board is mostly seen as \textit{append-only}, meaning that it is not possible to remove information from it or change anything that has been published \cite{37}. Furthermore only some of those involved in the system are able to write to it. The web bulletin board is also where the voter goes to check that her vote has been included in the final tally. It is also an important property of the web bulletin board that it displays the same information to anyone who reads information from it. Further work on the append-only web bulletin board can be found in \cite{37, 53}.

\textbf{Teller} A teller is a trusted party given the opportunity to take part in the running of the election system by performing a shuffle. The trust placed in a teller is not such that it would be able to change the outcome of the election. Instead the probability that at least one teller is honest is the mechanism safeguarding the secrecy of the election. Thus it is important to select a reasonable number of tellers such that they represent sufficiently diverse interests. One way of selecting tellers would thus

\footnote{Based on a personal feeling that technology is less liable to coercion than procedures (that is to say people).}
be to use a number of political parties: the probability that they would all work together to change the outcome of the election should therefore be small.

**Clerk** The clerks are involved in the setting up of the ballot forms. It may be beneficial to the security of the system to allow different organisations to be clerks and tellers, but in some cases it is possible for the tellers to fulfill the duties of the clerks without compromising the secrecy of the system.

**Encrypted receipt** The voter takes part in the creation of the encrypted receipt and thus knows that it encapsulates her vote. The voter is allowed to retain the receipt and after the close of the election she can check that it appears correctly on the web bulletin board, thus verifying the inclusion of her vote in the final tally. As the receipt is encrypted it does not reveal the vote it encapsulates and therefore cannot be used to coerce the voter or by the voter to prove to a vote buyer how she voted.

**Coercer** A coercer attempts to influence a voter to vote in a particular way (or spoil her vote or abstend from voting) by threat or reward. An electronic voting system is *coercion resistant* if it is possible for the voter to vote however she wishes without the coercer being able to check or find out whether she has complied with his request or not.

### 5.2 Simpler Prêt à Voter

#### 5.2.1 Motivation

Our work with the first Prêt à Voter implementation and the subsequent demonstrations have resulted in the identification of two main problems impeding the progress toward the running of a general election:

1. **OCR.** The Optical Character Recognition (OCR) used in the first version of the system was not very robust and in order to interpret the marks as successfully as possible, it required the voter to use a seven segment display (like those you see in LED clocks) and a thick pen. Although all agreed that the success rate of the OCR can be increased there was strong opposition from those with particular experience of implementing voting schemes against the seven segment display. It was felt that these were too cumbersome and hard to understand. We realise that this is not acceptable in a general election\(^3\) as such a voting system is used rarely by voters and this would introduce a large proportion of errors.

2. **Scanning.** The sheet-feed scanning of the ballot form is evidently very hard to use in elections where there are a number of races and/or a large number of candidates — election law may also stipulate that all races and candidates are printed on a single sheet, making this sheet immensely large. Furthermore, the layout of the

\(^3\)The first version of the implementation was intended as a working prototype.
ballot form would require that all candidates and their “boxes” were printed along the vertical axis of the paper, further limiting the number of races and candidates that can be printed on any piece of paper. Unfortunately, although that version of the Prêt à Voter implementation did support many concurrent different ballot forms it did not support the spanning of a single race over more than one ballot form.

The motivation for this configuration of Prêt à Voter is thus simplicity, accessibility and the accommodation of very large number of candidates. As the reader will see in the following sections this introduces some procedural safeguards where technological safeguards have previously been envisaged [86, 80]. We argue that this is not only necessary but that it is so important to include as many voters and introduce as few errors as possible in the voting process that the procedure/technology mix must be adjusted.

5.2.2 The voting ceremony

In the polling station there are a certain number of voting machines placed in voting booths. The secrecy of the election is based on these voting booths providing proper privacy to the voter and the voting machine similarly being unable to leak the intention of the voter. Thus there are poll station workers and guards keeping the area under surveillance in order to ensure that the machines cannot be tampered with.

The voter is able to enter the polling station without first identifying herself to the poll station staff and she can enter a voting booth so as to interact with the voting machine. It is important that she is not required to identify herself before she can interact with the machine because this makes it harder for the poll station staff or machine to connect the will expressed in the interaction with the machine to a particular voter.

The main purpose of the voting machine is to help the voter express her will in the election, the difficulty of which depends on the election system in place and the abilities of the voter. As the voter is interacting with a computer to make her choices the accessibility of the system is in itself an important area of research. It thus serves little use to go further into the details of how the voter interacts with the system to indicate her choices and it is sufficient to say that she may do so using her sight, touch and/or hearing and a touch screen, mouse, voice or other input device(s). At the end of the interaction the voting machine prints a vote in plain text (see Section 5.3.5) which the voter takes away and casts.

Interacting with the machine in the voting booth the voter is able to produce some number of votes. This must be a number greater than one so that the voter is able to create one vote that correctly captures her intention and some number of other votes that she can choose to audit, see below. Any malicious code running on the voting machine can therefore not determine whether a vote that is constructed on the machine

4Note that the accuracy is not threatened by this leak of information: but the privacy of the election is.

5This interaction is described below.
will be audited or if it will be cast. It should therefore be disinclined to attempt to alter the vote because there is some likelihood that it will be found out and the machine taken out of commission.

When the receipt is printed by the machine the voter can read it through and ensure that it is the vote she indicated to the machine. The vote she is going to cast she turns into an encrypted receipt by tearing it along the perforation. Any or all of the other votes she may have created she is able to have audited by approaching an auditing desk. The barcodes on these ballot forms are scanned in by poll station workers and the forms are decrypted and the information printed. The voter is now able to check that the printed information does correspond to the vote she has just audited. If so, she will grow more confident that the vote she will submit is also correctly formed.

Finally the voter approaches a submission desk with the encrypted receipt she wishes to submit. She identifies herself to poll station workers who check her inclusion in the list of eligible voters. The barcode on the encrypted receipt is scanned and the contents of it is electronically submitted to a central repository. Note that no submitted data must be kept secret to safeguard the secrecy of the election: it is already encrypted. After the close of the election, this, and all other encrypted receipts, will be decrypted as described in Section 5.3.8.

The voter can now leave the poll station with her encrypted receipt and after the close of the election she can use a website to check for the inclusion of her vote in the tally. She does this by entering the serial number of her encrypted receipt and comparing the image of the receipt served by the website with the actual receipt. If the marks on these match exactly she can be confident that her vote is included in the tally.

5.3 Technical foundation

5.3.1 Coping with Single Transferable Vote

In order to support Single Transferable Vote (STV) [104, 89] and other schemes where the voter expresses a ranking or awards votes to more than one candidate we employ the multiple-onion approach introduced by [36]. We provide an overview of the scheme here.

A numerical representation of a candidate is encrypted under a probabilistic threshold public key cryptography scheme. There are many different such encryptions for each candidate and as these are encrypted under a probabilistic scheme they do not look alike. We call these encryptions onions. A set of onions are associated with each ballot form and the voter’s choices, as expressed on the ballot form, are translated into an ordering of these onions. If the voter wishes to cast a vote for the candidates in the order $C$, $E$, $A$, $D$, $B$ then this is encoded by ordering the constituent onions thus:

$$O_C, O_E, O_A, O_D, O_B, O_{stop}$$

Note that these are encryptions and which candidate they represent is therefore hidden. The $stop$ onion $O_{stop}$ is used to ensure that the length of the vote is not dependent
on the number of choices expressed by the voter. A vote only for candidate C, for example, is thus constituted by an onion $O_C$, the stop onion and thereafter all other onions in a random order:

$$O_C, O_{stop}, O_A, O_E, O_D, O_B$$

After the close of the election the first constituent onion of each cast vote is decrypted and the vote given to the indicated candidate. This initiates the applicable STV protocol which removes candidates and redistributes the votes according to the next choice in order in a number of rounds until the required number of candidates have been elected. Each time the vote is redistributed the next choice is decrypted. In our example the first candidate is decrypted thus:

$$C, O_E, O_A, O_D, O_B, O_{stop}$$

If candidate C is subsequently eliminated and his or her votes redistributed, the onion representing candidate C is appended, the plaintext representation of C removed and the next onion decrypted, thus:

$$E, O_A, O_D, O_B, O_{stop}, O_C$$

This is now a vote for E. When a decryption reveals the stop onion the vote is removed from further redistributions. Each redistribution round contains a re-encryption shuffle so as to hide the ordering of the candidates in the vote, please see [36] for details. This configuration thus limits the impact of an attack popularly called the Italian attack [36] where the ordering of the candidates carries some message to a coercer.

### 5.3.2 Pre-creation of onions

A source of potential threats to the secrecy of the election pointed out in early papers describing end-to-end verifiable systems [76, 15, 82, 47, 83, 85, 84, 77, 78] was that the voting machine must select random values and errors or predictability in the pseudo-random number generator may render the cryptography useless. Furthermore, the voting machine might use “random” values from a list shared with a culprit or values such that a hash thereof would signal to a culprit the contents of the vote and/or the identity of the voter. To remove this problem we do not require the machine to select the randomness used in creating the candidate list.

The ballot form pre-creation process is started before the the election by a set of clerks (as described in [85]). In order to incorporate the STV strategy introduced above, the first clerk sets up two sets of onions, $O_L$ and $O_R$ (the candidates are listed in the base/alphabetical order) for each ballot form:

<table>
<thead>
<tr>
<th>$O_L$</th>
<th>$O_R$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$O_{LA}$</td>
<td>$O_{RA}$</td>
</tr>
<tr>
<td>$O_{LB}$</td>
<td>$O_{RB}$</td>
</tr>
<tr>
<td>$O_{LC}$</td>
<td>$O_{RC}$</td>
</tr>
<tr>
<td>$O_{LD}$</td>
<td>$O_{RD}$</td>
</tr>
<tr>
<td>$O_{LE}$</td>
<td>$O_{RE}$</td>
</tr>
</tbody>
</table>
$O_R$ is encrypted under the public threshold key of the tellers and $O_L$ under the public key of a particular voting machine (see below). Thus $O_{LA}$ and $O_{RA}$ contain the same plaintext but are encrypted under different public keys, meaning that the voting machine alone can decrypt $O_L$ and a threshold set of tellers must work together to decrypt $O_R$. The pair is denoted thus:

$$(E_{PK_M}(O_L), E_{PK_T}(O_R))$$

As a probabilistic encryption scheme such as Elgamal [26] or Paillier [65] is used, it is possible to change the appearance of the encryption by injecting randomness into the encryption without changing the plaintext. We take advantage of this property in the next step in which we wish to ensure that no-one knows the order of the candidates in $O_L$ or $O_R$.

The following procedure is now performed by each clerk in turn. One after the other, an onion pair $(O_L, O_R)$ is taken by the clerk from the web bulletin board. The order of the constituent onions is now permuted by the clerk based on a random value that it selects\(^6\). The clerk stores this random value in its internal database\(^7\) to be used for auditing at a later stage. For example the order of the constituent onions may now be:

<table>
<thead>
<tr>
<th>$O_L$</th>
<th>$O_R$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$O_{LC}$</td>
<td>$O_{RC}$</td>
</tr>
<tr>
<td>$O_{LB}$</td>
<td>$O_{RB}$</td>
</tr>
<tr>
<td>$O_{LE}$</td>
<td>$O_{RE}$</td>
</tr>
<tr>
<td>$O_{LD}$</td>
<td>$O_{RD}$</td>
</tr>
<tr>
<td>$O_{LA}$</td>
<td>$O_{RA}$</td>
</tr>
</tbody>
</table>

Note that each $O_{Lx}$ and $O_{Rx}$ still correspond. In order to hide the re-ordering performed by the clerk, it injects randomness into the encryption of each constituent onion. This randomness is also stored in the clerk’s internal database. This process is called to re-encrypt the onions and results in a set of constituent onions that cannot be related to the previous set without first performing a decryption, in our example:

<table>
<thead>
<tr>
<th>$O'_L$</th>
<th>$O'_R$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$O'_{LC}$</td>
<td>$O'_{RC}$</td>
</tr>
<tr>
<td>$O'_{LB}$</td>
<td>$O'_{RB}$</td>
</tr>
<tr>
<td>$O'_{LE}$</td>
<td>$O'_{RE}$</td>
</tr>
<tr>
<td>$O'_{LD}$</td>
<td>$O'_{RD}$</td>
</tr>
<tr>
<td>$O'_{LA}$</td>
<td>$O'_{RA}$</td>
</tr>
</tbody>
</table>

The clerk now writes $(O'_L, O'_R)$ to a new column, its middle column, on the web bulletin board. The process is repeated on $(O'_L, O'_R)$ and $(O''_L, O''_R)$ is written to a third column, the clerk’s output column.

---

\(^6\) The clerk can choose this value however it wants: the secrecy of the system is based on a number of clerks performing this operation and thus if at least one clerk is honest the secrecy is safeguarded.

\(^7\) Each clerk must keep its database secret, but the secrecy of the system is only jeopardised if all clerks leak their databases.
5.3. Technical foundation

<table>
<thead>
<tr>
<th>$N^*$</th>
<th>Setup</th>
<th>Clerk 1</th>
<th>Clerk 2</th>
<th>Clerk 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$(O_L, O_R)$</td>
<td>$(O'_L, O'_R)$</td>
<td>$(O''_L, O''_R)$</td>
<td>$(O'''_L, O'''_R)$</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>$(O''''_L, O''''_R)$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td>$(O''''''_L, O''''''_R)$</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$(O''''''_L, O''''''_R)$</td>
<td></td>
</tr>
</tbody>
</table>

Table 5.1: The pre-creation web bulletin board

The whole process is now repeated by each clerk in order for all $(O_L, O_R)$ pairs on the web bulletin board, as shown in Table 5.1. The result of this process is a final column containing $(O_L, O_R)$ pairs where the order of the constituent onions are unknown to all parties and can only be found out if all clerks collude or a decryption is performed.

Note that the initial creation of the onions and the re-encryptions of the same must be audited to ensure that the clerks do not break the link between two onions by injecting different random values into the plaintext. Here we do this by randomly auditing complete re-encryption chains but it may also be possible to do with zero-knowledge proofs, as done in [85].

5.3.3 On-demand distribution

As stated previously the purpose of creating the onions before the election is to ensure that no single entity has undue influence over their creation. In order to make use of the onions, however, they must be distributed to the polling stations and voting machines. The foundation of this distribution is the assignment of a private key to each electronic voting machine. It follows logically that this private (secret) key must be loaded into the machine and it must be ensured that it cannot be extracted from the machine. It is thus plain that the secrecy of the election is based on safe and secure creation, distribution and use of private/public key pairs. This is normally called a Public Key Infrastructure (PKI). Furthermore the voting machines may require the use of some elements of tamper-proof hardware to ensure that no covert channels, leaking the choices made by the voter (very much like a camera in the voting booth would), exist. We do not cover these topics here.

As discussed in the introduction, we are here opting for a procedure/technology mix with higher emphasis on procedural solutions than may have been employed in previous schemes. One such procedural solution is the use of a PKI as described above.

When the voter interacts with the voting machine (see next section) the machine requires the $O_L$ set of onions to produce a vote. There are two ways of distributing these onions to the voting machine: either using a network (Internet) connection or by pre-loading a batch of onions.

**Network connection**

If the voting machine is equipped with a network connection and it is able to connect to a central repository over this network, it can fetch the required onions on demand.
Chapter 5. Simple and secure electronic voting with Prêt à Voter

The main disadvantage to this approach is that the network connection must be made as secure as possible but can be used to carry covert communication by a tampered/infected machine to an accomplice. As the running of the election is based on the availability of the network and the servers on this network this makes the system arguably fragile.

The main advantage to this on-demand distribution is that if the clerks are online during the election phase they are able to create any number of onions as required by the machines. This means that only the exact number of onions required has to be created which reduces storage demands.

**Pre-loading of batch**

Before the election, as described in the previous section, the clerks create a certain, large, number of onions for each voting machine. These onions are then stored on some mobile media (hard drive, flash drive, CD or DVD etc) and transported to the polling stations and loaded into the appropriate machines. The foremost advantage of this setup is that no network is required to run the election, eliminating an arguably very fragile part of the system. Furthermore, if no network is required then the machine in the voting booth becomes fully autonomous.

The main disadvantage of course is that a very large number of onions must be created by the clerks in advance and distributed “by hand” and when a voting machine runs out of onions it may be a lengthy process to replenish its store.

We include both these options here to show that the system is quite adaptable to the particular circumstances under which it is used. Note that the voting machine has no need to communicate with the outside world, only to receive prepared onions from the clerks.

**5.3.4 Touch screen interface**

To accommodate for elections with many races and/or races with many candidates the proposed configuration of Prêt à Voter has two major differences to previous versions: (a) the receipt is created by a voting machine and (b) the receipt is printed in the minimal form presented in the next section.

**Preparing the machine**

Before the voting machine can be used it must be loaded with the definition of the election, its races and their candidates. This is a detail of the implementation and as such is not defined in detail here: the scheme presented here only requires the machine to be loaded with this information in advance of its use. This means this scheme is compatible with many different scenarios, from a single election with a number of races where all voters are able to cast (a) vote(s) to constituency based voting where a voter may only be eligible to vote in a subset of all the defined races.

---

8 After performing audits to ensure correctly formed onions.
9 Please note that we have not made an implementation of this system at the time of writing.
Creating a vote with the machine

This is an example of a possible interaction with the voting machine. The steps involved can be different in appearance, order and number and are adapted to the election.

Approaching an idle voting machine, the voter is greeted with a message asking her to touch the screen to initiate the voting process.

| Springfield Local Election | Tap screen to start |

A list of races is shown with indicators to whether or not a vote has been created in each race. The voter selects a race by tapping the screen.\(^\text{10}\).

<table>
<thead>
<tr>
<th>Select race</th>
<th>Not voted</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mayor</td>
<td>Not voted</td>
</tr>
<tr>
<td>Sanitation Commissioner</td>
<td>Not voted</td>
</tr>
</tbody>
</table>

A list of the candidates in the selected race is shown and the voter is able to tap a single candidate or a number of candidates in the preferred order. A “Clear” button is available on the screen, which clears all choices made and allows the voter to start over. A “Proceed” button allows the voter to return to the list of races.

<table>
<thead>
<tr>
<th>Vote for Sanitation Commissioner</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shmoikel Krustofsky</td>
</tr>
<tr>
<td>Apu Nahasapeemapetilon</td>
</tr>
<tr>
<td>Ray Patterson</td>
</tr>
<tr>
<td>Homer Simpson</td>
</tr>
</tbody>
</table>

Selecting her favourite candidate, the voter completes the vote for the race and clicks the “Proceed” button to return to the race selection screen.

| Select race | Not voted | Voted |
|-------------|-----------|
| Mayor       | Not voted |
| Sanitation Commissioner | Voted |

The voter is able to return to any race and re-create her vote. A “Proceed” button on the race selection screen allows her to go to a summary screen. Here the voter can select either of two buttons: “Go back” or “Print vote”.

<table>
<thead>
<tr>
<th>Summary of your vote</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mayor</td>
</tr>
<tr>
<td>Sanitation Commissioner</td>
</tr>
</tbody>
</table>

When the voter is finished and presses the “Print vote” button, the machine displays a final message whilst printing the vote (shown in the next section).

<table>
<thead>
<tr>
<th>Thank you</th>
</tr>
</thead>
<tbody>
<tr>
<td>Please take your printed vote</td>
</tr>
</tbody>
</table>

\(^{10}\) Or using some other input method, depending on the abilities of the voter.
5.3.5 The minimalistic encrypted receipt

The purpose of the minimalistic encrypted receipt is to enable the printing of many races on the same receipt and to aid the voter in checking the receipt on the web bulletin board. To achieve this we wish to print as few candidates as possible on the receipt. We first introduce the traditional Prêt à Voter ballot form and its associated encrypted receipt before showing the alterations we propose to these.

The Prêt à Voter ballot form and encrypted receipt

The ballot form in Prêt à Voter consists of two columns: in the left the candidates are printed in a random order (based on randomness unique for the form) and in the right the voter makes her marks in a grid corresponding to the candidates in the left column. For example:

<table>
<thead>
<tr>
<th>Ballot form</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sanitation Commissioner</td>
</tr>
<tr>
<td>Homer Simpson</td>
</tr>
<tr>
<td>STOP</td>
</tr>
<tr>
<td>Apu Nahasapeemapetilon</td>
</tr>
<tr>
<td>Ray Patterson</td>
</tr>
<tr>
<td>Shmoikel Krustofsky</td>
</tr>
</tbody>
</table>

If a voter makes her marks in the right hand side grid and then detaches and destroys the left hand column, the remaining encrypted receipt does not reveal her vote. However, a value called the onion, printed at the bottom of the grid, can be decrypted to reveal the vote. In this example an encrypted receipt may be:

```
2
3
1
lk3j92784
```

It has been envisaged that the Prêt à Voter ballot form is a single page, which contains all races in the election and all the candidates in each of those races. The voter makes her mark on the paper and detaches and destroys half, producing an encrypted receipt which is subsequently scanned and then handled electronically. It is quite clear that in an election with many races and many candidates, it is not possible to print all on one piece of paper that can also be fed through a scanner after the marks have been made by the voter.
The minimalistic encrypted receipt

The traditional Prêt à Voter ballot form is printed onto paper before the election (or on demand before they are used [85, 54]) and as the voter uses a pen to fill out her choices, naturally all candidates must be available on the ballot form. In the scheme presented here a computer is used to create the vote after which the ballot form is printed. Therefore, it is possible to print only the candidate(s) that the voter has indicated a vote for.

In our example, when the voter makes her marks using the touch screen she may indicate her choices thus (note that the candidates are listed in the alphabetical order on the screen):

<table>
<thead>
<tr>
<th>Vote for Sanitation Commissioner</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shmoikel Krustofsky</td>
</tr>
<tr>
<td>Apu Nahasapeemapetilon</td>
</tr>
<tr>
<td>Ray Patterson</td>
</tr>
<tr>
<td>Homer Simpson</td>
</tr>
</tbody>
</table>

When the voter presses the “Print receipt” button the voting machine retrieves the necessary onions and decrypts these (see above) to find the ordering of the candidates. Let us assume in our example that the machine retrieves the onions with serial number 27344, decrypts these and finds that the candidate list has the following order:

<table>
<thead>
<tr>
<th>27344</th>
</tr>
</thead>
<tbody>
<tr>
<td>Homer Simpson</td>
</tr>
<tr>
<td>STOP</td>
</tr>
<tr>
<td>Apu Nahasapeemapetilon</td>
</tr>
<tr>
<td>Ray Patterson</td>
</tr>
<tr>
<td>Shmoikel Krustofsky</td>
</tr>
</tbody>
</table>

The machine now prints the following filled-out Prêt à Voter ballot form, note that only the candidates which the voter has indicated are printed and that these are printed in the order dictated by the onions:

<table>
<thead>
<tr>
<th>Ballot form</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sanitation Commissioner</td>
</tr>
<tr>
<td>Homer Simpson</td>
</tr>
<tr>
<td>STOP</td>
</tr>
<tr>
<td>Ray Patterson</td>
</tr>
<tr>
<td>1, 2, 4</td>
</tr>
<tr>
<td>27344</td>
</tr>
</tbody>
</table>
27344. These values can be printed in the form of a barcode (see below) which allows them to be read in quickly. Note that these numbers together with the choices indicated above by the voter is all that is needed to represent the vote.

The voter now checks that the printed vote is truly a representation of her intended vote, simply by reading the candidates’ names and the number assigned to each. If the vote is not correctly printed she can discard the it (by shredding it for example) and produce another\(^{11}\). If she is happy with the vote and wishes to cast it, she detaches the two columns from each other and destroys the left hand one. What remains is an encrypted receipt:

\[
\begin{array}{c}
2 \\
3 \\
1 \\
1, 2, 4 \\
27344 \\
\end{array}
\]

The voter approaches a desk manned by poll station staff, identifies herself and allows the barcode on the encrypted receipt to be scanned. When poll station staff are satisfied that the barcode has been scanned and electronically transmitted to the web bulletin board they stamp the encrypted receipt with an official stamp so as to indicate that it is the receipt of a vote that has been cast in the election. A mark is placed in the register to indicate that this voter has cast her vote\(^{12}\).

All votes submitted in this way are collected on the web bulletin board.

The barcode

All previous versions of Prêt à Voter has required an encrypted receipt to be scanned in and interpreted to form a digital representation that could subsequently be decrypted. This OCR process has been shown to be a significant weakness to the scheme: it results in many errors\(^{13}\).

In this scheme we reduce the amount of work in the scanning process to the recognition of a barcode. These are printed in such a way as to be simple to read and recognise and they can contain check numbers etc to aid the correct interpretation of them.

In order to record a vote the system must read the following information from the encrypted receipt:

\(^{11}\)The ability to print a vote in plain text on the machine and check that it is correct serves to audit that the machine works correctly [9]. The voter is thus free to print any number of votes, either with the choices she truly wants to vote for or, if she is actively auditing the machine, for other candidates. If the machine prints the vote incorrectly the voter may be discouraged from alerting officials if she has to show a “real” printed vote. Furthermore, as no proof exists of what instructions the voter has given the machine, raising an alarm if the machine prints an incorrect vote is hard.

\(^{12}\)In some places, such as the United Kingdom, the law requires that the ballot form serial number is noted against the name of the voter: that is quite possible to do in this scheme.

\(^{13}\)Note that these errors did not mean that a vote was cast for a different candidate than indicated by the voter — but that the vote had to fill out another ballot form as the first could not be correctly understood by the system.
5.3. Technical foundation

1. The serial number (27344)

2. Which candidates are shown on the ballot form (1, 2, 4)

3. The marks made by the voter (2, 3, 1)

To enter this information into the barcode, we simply concatenate them:

27344|1, 2, 4|2, 3, 1

When this information is scanned by poll station staff it is submitted to the web bulletin board. Here the appropriate constituent onions are retrieved:

```
+-----------------+
| 27344           |
+-----------------+
| RSimpson        |
| RSTOP           |
| RNahasapeemapetilon |
| RPatterson      |
| RKrustofsky     |
```

The appropriate onions are selected (numbers 1, 2 and 4 in our example) and reordered in the correct order as indicated by the choices (2, 3 and 1) — thus the onions are placed in the following order:

```
+-----------------+
| 27344           |
+-----------------+
| RPatterson      |
| RSimpson        |
| RSTOP           |
```

Note that of course the contents of these onions are unknown! Therefore the system now holds an encrypted vote submitted by this voter.

5.3.6 Auditing a vote

We here argue that it is safe to allow a voter to use a voting machine to create the vote, because she may create any number of votes and audit some of these. If the voting machine attempts to cheat it cannot be sure that the vote will not be audited and its cheating thus found out. A malfunctioning machine will thus be found with a high probability and taken out of commission.

The first audit that a voter does of a vote printed by the voting machine is simply to read it. If the machine has committed an error (or something worse) then the marks printed would not match the intention of the voter. If this is the case she can simply destroy the vote and create another one — until she receives one that correctly indicates the vote she wishes to cast. Note that the voter may have performed some “human” error while interacting with the machine and not spotted this until the vote has been printed: this gives her another chance to spot such a mistake and to rectify it.
The second audit of the ballot form that can be performed on any vote is the checking of the barcode. This is simply done by the voter allowing the barcode to be scanned by a machine available in the polling station which shows the contents of the barcode in a human readable form. Such machines can also be supplied by independent organisations. The voter then simply checks that the information shown by the reader corresponds to the information printed in the right column of her vote.

Finally, if the voter decides to audit a created vote\footnote{Note that the voter can print any number of votes and therefore she can create a vote with completely different choices to the vote she subsequently intends to cast.} then the constituent onions $O_R$ shall be retrieved from the web bulletin board (where they are marked as audited, ensuring that no vote can subsequently be cast with these onions) and decrypted by the tellers. The full candidate list is then displayed to the voter who compares it to the printed vote.

The purpose of this audit is first to find any machine that may malfunction or that has been compromised. Secondly the audit functions to convince voters that the system is working correctly and that the vote will be decrypted correctly.

\subsection*{5.3.7 Checking the receipt}

The voter is allowed to take home the scanned and stamped encrypted receipt. She can then, at any time, visit the web bulletin board on the web and search for the serial number printed on the receipt. When she calls up her receipt she should see an exact replica of the receipt she holds in her hands. If this is the case then the voter can be certain that her vote has been included in the final tally. If the receipt is not found on the web bulletin board or if the version she finds there does not match the one she has in her hand, she can accuse those in charge of running the election of malfunction or fraud and she has proof in her receipt that she has cast a vote which is now missing or has been changed.

\subsection*{5.3.8 Decryption and tallying}

At this stage the web bulletin board contains a list of all encrypted votes that have been cast, in the form of a number of ordered onions. A detailed specification of how the decryption is performed is available in \cite{36} but we provide an overview here.

As shown in the pre-creation of the onions (Section 5.3.2) the appearance of the onions can be changed without altering the plaintext hidden within. We make use of this same method to break the link between the encrypted receipt (and thus the voter) and the plaintext vote. This is achieved by allowing the tellers to perform a number of re-encryption mixes before decrypting the onions and starting the STV protocol.

An example re-encryption mix network for a mere three votes is shown in Table 5.2. Each teller reads all votes on the web bulletin board as a batch, performs two re-encryption mixes and writes each of the resulting batches to the web bulletin board. When a threshold set of the tellers have satisfied themselves that the votes are thoroughly
mixed, they perform the threshold encryption of the first constituent onion of all votes in the final batch. These partially decrypted votes go into the first round of the STV algorithm.

### 5.3.9 Note on securing the machine using procedures

It is important to note that the accuracy of the election, that is to say the trustworthiness of the outcome of the election, is safeguarded not by procedures but by the cryptographic properties of the system. The result of the election is thus as trustworthy as in previous configurations of Prêt à Voter [20, 85], because they all rely on the same verifiability.

### 5.4 Summary

In this chapter we have introduced a variant of Prêt à Voter that uses a touch screen type interface to help the voter form her vote. This is a possible complement to the normal paper based voting in the original system that aims to be accessible for those unable to cast a paper ballot. We have also shown the minimalistic encrypted receipt, a method of printing a Prêt à Voter style encrypted receipt that aims to solve problems with races with a large number of candidates and OCR issues.

### 5.5 Discussion

The main advantages of the proposed scheme is that the voting machine is able to guide the voter through a potentially very complex voting procedure involving any number of races and any number of candidates in those races. The voter turns the plain text vote into an encrypted receipt and the scanning of this receipt is very fast because only a barcode has to be scanned.
The main disadvantage to this configuration of Prêt à Voter is that the voting machine must learn the voter’s intention in order to produce the receipt. The secrecy of the election is thus safeguarded simply by procedures that ensure that the machine does not leak any information. As discussed in the introductory sections of this paper there is a necessity to alter the procedure/technology mix so that it is possible to make the system more accessible and remove a large proportion of the errors associated with the filling out of the ballot form.
6 A human readable paper audit trail

6.1 Introduction

Electronic voting machines have, because of their dependability on correctly written code and the correct execution of (human) processes, become heavily criticised, mainly for failures that have occurred in elections in the USA during the first decade of the 21st century. Such direct recording equipment (DRE) that lack the verifiability of the systems discussed in this thesis are liable to manipulation without detection [48].

A measure that has been proposed as an extension to already existing machines is the Voter Verifiable Paper Audit Trail (VVPAT) [58]. This is a procedure whereby the DRE prints the vote in a human-readable form behind a sheet of glass and displays it to the voter in the voting booth. This allows the voter to check that the vote is correctly formed and when this check is complete the printed vote is automatically dropped into a sealed ballot box. If the voting system is subsequently challenged, the printed votes in the box can be counted and compared to the electronically tallied result.

However, paper is not necessarily a simple solution. There are problems, such as:

- Paper audit trails are not invulnerable to corruption.
- It is not clear how any conflicts between the computer and paper audit counts should be resolved.
- Humans are notoriously bad at proof-reading, especially their own material, and hence bad at detecting errors in a record of their choices [24].
- Even if the voter does notice a discrepancy with the paper record created at the time of casting, it may be tricky to resolve, especially without undermining the privacy of the ballot.
- It is not clear under what circumstances the audit trail should be invoked.

Verifiable schemes such as Prêt à Voter (Section 2.4.2), VoteHere [61], and PunchScan [28], provide higher levels of assurance than even conventional paper based elections and certainly far higher assurance than systems that are dependent on the correctness of (often proprietary) code. However, the arguments as to why the systems offer high assurance are highly technical (mathematical) and therefore it seems probable that the general public must trust experts to examine the system and it is yet unclear if they do trust experts in this way. During trials of verifiable systems it may therefore be beneficial to add conventional auditing mechanisms to support public confidence.
One possible path to common understanding of the system is supplementing a cryptographic scheme with a conventional paper audit trail backup that we refer to as a Human Readable Paper Audit Trail (HRPAT). The addition of a HRPAT to Prêt à Voter was first described in [79], which we here extend so as to achieve several properties (described below) in addition to the fundamental verifiability property of the HRPAT.

As discussed by Ryan in [79], introducing a mechanism such as a HRPAT may introduce certain vulnerabilities not present in the original scheme. However, it may be argued that it is worth introducing such risks, at least during trials and early phases of deployment, with the view to remove the paper trail when the system has “proven itself”.

This chapter details some recently proposed changes to [79] that gives rise to a number of additional auditing possibilities. This maximises the reassurance of having a conventional mechanism as a backup while minimising threats to ballot privacy. Once sufficient levels of trust and confidence have been established in a verifiable, trustworthy scheme like Prêt à Voter, we would hope that the scaffolding of an HRPAT could be cast aside.

The scheme presented here extends Ryan’s scheme with a number of technical benefits (besides the confidence building aspects). It can provide a robust counter to the danger of voters attempting to leave the polling station with the left hand part of the Prêt à Voter ballot form. This shows the candidate order and so could provide a potential coercer with proof of the vote. A number of possible counter-measures to this threat have been identified previously, for example the provision of decoy candidate lists [82, 84], but the mechanism here appears to be particularly robust. The procedure we propose here involves the officials verifying that the voter submits the component of the ballot that carries the candidate order at the time of casting.

The approach proposed here enables a number of additional auditing procedures to be introduced that significantly increase the assurance of accuracy, assuming that the integrity of the paper audit trail can be ensured. We also show how the integrity of the human readable paper audit trail can be verified in this scheme, something safeguarded only by procedures in previous schemes.

6.2 The scheme

This section presents the HRPAT Prêt à Voter ballot form with its onions and how they are created and printed. Then the on-demand printing of the candidate list and the casting of ballots is described. The last part of the section details the decryption of the encrypted receipts and how the HRPAT is used to verify the electronic election.

6.2.1 The ballot form and its use

The usual Prêt à Voter ballot form is modified to comprise two overlaid pages. The bottom page has the usual two portions: the left hand portion carries an onion and a serial number. The top page overlays the right portion of the bottom sheet and carries
### 6.2. The scheme

<table>
<thead>
<tr>
<th>POST</th>
<th>RETAIN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>onion&lt;sub&gt;L&lt;/sub&gt;</td>
<td>onion&lt;sub&gt;R&lt;/sub&gt;</td>
</tr>
<tr>
<td>serial</td>
<td></td>
</tr>
</tbody>
</table>

**Figure 6.1:** The ballot form in two pages

<table>
<thead>
<tr>
<th>RETAIN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td>onion&lt;sub&gt;L&lt;/sub&gt;</td>
</tr>
<tr>
<td>serial</td>
</tr>
</tbody>
</table>

**Figure 6.2:** The ballot form complete

<table>
<thead>
<tr>
<th>RETAIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>candidate&lt;sub&gt;B&lt;/sub&gt;</td>
</tr>
<tr>
<td>candidate&lt;sub&gt;C&lt;/sub&gt;</td>
</tr>
<tr>
<td>candidate&lt;sub&gt;A&lt;/sub&gt;</td>
</tr>
<tr>
<td>onion&lt;sub&gt;L&lt;/sub&gt;</td>
</tr>
<tr>
<td>serial</td>
</tr>
</tbody>
</table>

**Figure 6.3:** The ballot form with candidates printed

<table>
<thead>
<tr>
<th>RETAIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>candidate&lt;sub&gt;B&lt;/sub&gt;</td>
</tr>
<tr>
<td>candidate&lt;sub&gt;C&lt;/sub&gt;</td>
</tr>
<tr>
<td>candidate&lt;sub&gt;A&lt;/sub&gt;</td>
</tr>
<tr>
<td>onion&lt;sub&gt;L&lt;/sub&gt;</td>
</tr>
<tr>
<td>serial</td>
</tr>
</tbody>
</table>

**Figure 6.4:** The ballot form with marks

<table>
<thead>
<tr>
<th>POST</th>
<th>RETAIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>candidate&lt;sub&gt;B&lt;/sub&gt;</td>
<td></td>
</tr>
<tr>
<td>candidate&lt;sub&gt;C&lt;/sub&gt;</td>
<td>X</td>
</tr>
<tr>
<td>candidate&lt;sub&gt;A&lt;/sub&gt;</td>
<td></td>
</tr>
<tr>
<td>onion&lt;sub&gt;L&lt;/sub&gt;</td>
<td>onion&lt;sub&gt;R&lt;/sub&gt;</td>
</tr>
<tr>
<td>serial</td>
<td></td>
</tr>
</tbody>
</table>

**Figure 6.5:** The marked ballot form in two pages
another onion value. The top page has a carbon layer or similar on the back to ensure\(^1\) that marks applied to the top page transfer to the bottom page. The layout of the ballot form is shown in Figure 6.1. This means that when the top page is aligned over the right column of the bottom page, as is the case when the voter receives the ballot form, the ballot form looks as shown in Figure 6.2. When the voter makes her mark in the right hand column of this complete form the mark is made on both pages.

The reader will notice that there are no candidate names printed in Figure 6.1. This is because we are incorporating the on-demand printing of ballot forms introduced in\(^2\) [85]. When the voter has identified herself to the poll station workers she is allowed to randomly choose a ballot form such as that in Figure 6.2. At this stage onion\(_{L}\) and onion\(_{R}\) are concealed (for example by a scratch strip) so that they cannot be read by either the poll station worker nor anyone else at the polling station. The other value, serial, is noted in the register next to the voter’s name.

The voter takes the form into the voting booth where she makes onion\(_{L}\) visible and then allows a machine to read this value. The machine decrypts the onion, as will be explained later, and from this computes the candidate list, which it now prints in the left column of the ballot form. The result is depicted in Figure 6.3.

The voter now makes her mark(s) on the form in the privacy of the voting booth and the result is exemplified in Figure 6.4. She then detaches the top page from the bottom and the result is shown in Figure 6.5. The voter places the page marked POST into an envelope through which only the serial number is visible and then leaves the booth carrying the envelope and the top page, which will constitute her receipt. She now presents herself to the vote casting desk and hands over the envelope and receipt. The poll station worker checks that serial is the same as the one previously assigned to the voter. Once this is done, the serial number is detached and discarded and the envelope containing the lower page is placed in the ballot box. The page marked RETAIN, which acts like a conventional Prêt à Voter receipt, is scanned, a digital copy posted to the WBB and handed back to the the voter to keep as her protected receipt.

The serial number serves a dual purpose here: firstly it counters chain-voting attacks as suggested by Jones [45]. Secondly, it serves to verify that the voter does not retain the lower layer of their ballot form. This is a useful spin-off of the HRPAT mechanism: in the standard Prêt à Voter, there is the possibility of the voter retaining the left-hand (LH) portion of the ballot form, along with her receipt, to prove to a coercer how she voted\(^3\).

### 6.2.2 Cut-and-choose

Early versions of Prêt à Voter used preprinted ballot forms and so, for the election to be guaranteed accurate and to instil trust in the voters, randomly selected ballot forms

---

\(^1\)See Section 6.2.5 for details on resolving any disputes about this.

\(^2\)It is possible to alter the methods described here and use pre-printed ballot forms — fulfilling a lower secrecy level.

\(^3\)Although procedures may be in place to stop voters from doing this it is very hard to do so without violating vote secrecy.
are audited before, during and after the election. That is to say they are decrypted and shown to have been correctly printed [21, 82]. Such random selection is performed by suitable auditing authorities but may also be supplemented by the voters themselves. One mechanism to provide such a cut-and-choose protocol to the voter while maintaining control on the number of ballots issued to each voter, is to have a double sided form, one side of which (selected at random by the voter) is used to cast the vote and the other is automatically audited [84, 85]. However, any such “cut-and-choose” mechanism only allows forms that are not used to be audited.

In the scheme presented here, we add a paper audit trail to Prêt à Voter. As has been described above, the candidate list is printed on the bottom page of the ballot form and this page is placed in a ballot box and provides the human readable paper audit trail. Because of the properties of the relation between the two pages as described in this section, it is possible to audit the printing of the candidate list of any number of forms, that were actually used for voting, after the close of the election. The device or authority printing the form would thus be caught with a probability proportional to the number of forms audited. Hence the HRPAT method shown in this section has this further audit application. This auditing mechanism can be used with either pre-printed or on-demand printed forms.

### 6.2.3 Generation of the encrypted ballot forms

We describe a distributed, parallel construction of the onion pairs, analogous to the Paillier construction presented in [80]. Suppose that we have \( L \) clerks. They will be responsible for generating \( I \) onion pairs, where each onion pair will carry the same seed/plainext.

We further suppose that we have an Elgamal public key for the tellers \( PK_T \) and public keys for the booths \( PK_B, \) where \( k \) indexes the booths. Both of these public keys will have the same modulus. We provide the construction for a single booth key; we simply replicate the construction for other booth keys. Denote the public key of the booth in question as \( PK_B. \)

The \( j \)th clerk generates \( I \) sub-onion pairs:

\[
\{\theta^T_{j,i}; \theta^B_{j,i}\}
\]

Where:

\[
\theta^T_{j,i} := \{s_{j,i}, x_{j,i}\}_PK_T
\]

and

\[
\theta^B_{j,i} := \{s_{j,i}, y_{j,i}\}_PK_B
\]

The first term is an encryption of the \( j, i \)th seed under the teller’s public key. The second term is the encryption of the same seed value under the booth’s public key. The randomisations \( x, y \), used for these two encryptions should be independent.
All of these sub-onions are all posted to a WBB in cells of an $L \times I$ matrix ($L$ columns, $I$ rows) — one pair in each cell. To audit these, an independent auditing entity chooses for each row a randomly selected subset of the cells in the row, say half. For these selected cells the clerks reveal the $s$, $x$ and $y$ values. The auditor can check that the encryptions match the posted sub-onion values and that the two seed values are equal for each pair. The auditor can also check that the $s$ values are consistent with the required distribution.

Assuming the posted material passes the audits, the “full” onions are formed by taking the product of the remaining, un-audited pairs row-wise. This step is universally verifiable. Let $A_i$ denote the set of indices of the pairs selected for audit in the $i$th row. Then the “full” onions for the $i$th row are computed as:

$$\Theta^T_i := \prod_{j \in A} q^T_{j,i}$$

$$\Theta^B_i := \prod_{j \in A} q^B_{j,i}$$

To create the proto-ballots, suppose that we have paper ballot forms that initially just carry index values from $I$, each form will carry a unique index value. We now introduce two new processes $P_1$, $P_2$. $P_1$ takes a form with index $i$, looks up $\Theta^T_i$ on the WBB, re-encrypts it and prints the result on the right-hand (RH) portion of form. This now constitutes the $\Theta_{R,i}$ for the ballot form. It then covers this with a scratch strip. Once it has finished a batch of these, they are shuffled and passed on to $P_2$.

$P_2$ looks up the appropriate $\Theta^B_i$, re-encrypts this and prints the resulting value, $\Theta^B_{L,i}$ on the LH portion of the ballot and covers it with a scratch strip.

We perform audits on a randomly selected subset of the resulting proto-ballots. For the selected ballots, the onions are revealed and $P_1$ and $P_2$ are required to prove the re-encryption link back to the onion pair on the WBB. Audited forms are discarded.

Our construction ensures that it would take a corrupt booth or access to the paper audit trail and a two-way collusion, of $P_1$ and $P_2$, to link the $R$ (receipt) onions to the candidate lists. The index value on the ballots can serve as the serial number, and is removed at the time of casting.

### 6.2.4 Anonymising Tabulation

Anonymising tabulation proceeds as for Prêt à Voter 2006 (Section 2.4.2). We outline it here for completeness. The encrypted receipts scanned in the polling station are published on the web bulletin board and all voters are able to check that their receipts appear there. When all tellers are satisfied that the election has ended and all electoral rules have been followed they start the decryption process, which is shown in Table 6.1. The first teller, $T_1$, takes all encrypted receipts and injects the voter’s choice(s) into the onion$_R$, using the homomorphic properties of exponential Elgamal. We call the onion with the injected choice(s) onion$_I$. Suppose:
Table 6.1: Decryption of the encrypted receipts

\[
onion_R = (\alpha^r, \alpha^s \cdot \beta^r) \pmod{p}
\]

Then:

\[
onion_I := (\alpha^r, \alpha^v \cdot \alpha^m \cdot \beta^r) \pmod{p}
\]

The index number \(v\) indicates the position of the \(X\) on the receipt. In effect, we are multiplying \(\onion_R\) by the encryption of \(v\) with randomisation \(r = 0\). The result is:

\[
onion_I = \{v + s, t\}_{PK_T}
\]

Thus, the \(I\) onion is the encryption of the \(v\) index plus the seed value. The offset \(\phi\) of the candidate list printed on the ballot form is computed as \(\phi := s \pmod{n}\), where \(n\) is the number of candidates. The candidate order is cyclically shifted upwards from the canonical ordering by \(\phi\). Thus, \(v + s \pmod{n}\) gives the index of the candidate chosen by the voter in the canonical ordering of the candidates.

No mixing is performed at this step: the \(I\) and \(R\) onions are posted side-by-side on the WBB. That each \(\onion_I\) is correctly formed with respect to \(\onion_R\) is thus universally verifiable.

A set of mix tellers now perform a sequence of re-encryption mixes. Each mix teller takes the full batch of \(\onion_I\)s, re-encrypts each onion, shuffles the batch and outputs to the next mix teller. The output batch from each teller is published onto the web bulletin board. The last output batch we call \(\onion_{In}\).

When all mix tellers have performed their re-encryption mixes, the independent auditors confirm that the mixes have all been performed correctly. This might be done using partial random checking [44], or perhaps Neff’s proofs of Elgamal shuffles [62]. If the auditors confirm that the mixes are correct, we can proceed to the decryption stage. If problems are identified with the mixes, corrective actions can be taken. Thus, for example, if one of the mix tellers is identified as having cheated, it can be removed and replaced. The mixes can be re-computed from that point onwards and re-audited. We might routinely re-run the mixes and audits, in any case, for additional assurance.

Once we are happy that the mixes have been performed correctly, a threshold set of the decryption tellers take over and cooperate to decrypt each \(\onion_{In}\). No mixing is
required at this stage and each step of the decryption can be accompanied with a ZK proof of correct (partial) decryption. The final, fully decrypted values can be translated into the corresponding candidate values using:

\[ \text{candidate}_i = (s + v) \pmod{n} \]

Such re-encryption mixes are known to provide anonymity against a passive attacker. Against an active attacker, who might have some capability to inject or alter terms entered into the mix, we have to guard against ballot doubling attacks: to identify a particular voter’s choice, he injects a term that is a re-randomisation of the voter’s receipt. If unchecked, this will result in two decrypted receipts with the same adjusted seed value. We will in any case have procedures in place to guard against ballot stuffing that will help counter such dangers. An additional measure is to run (threshold) PETs (plaintext equivalence tests) against the terms in the mix prior to decryption (see [46]).

### 6.2.5 Audit of the paper trail

We now have a number of possible strategies for auditing the election. One scenario is to perform a full, manual recount of the election using the HRPAT and simply compare this with the cryptographic count. In practice, due to inevitable errors with manual counting, this will differ from the electronic count, even if the latter is exact and correct. If the difference is small and well within the winning margin, this could probably be disregarded.

An alternative is to take a random subset of the HRPAT ballots and, for each of these forms, the auditor requires the appropriate booth to decrypt the onion and so reveal the seed \( s \). The tellers are required to provide ZK proofs of the correctness of their decryption steps. From the seed value \( s \) it computes the candidate order and checks that this agrees with the list printed on the ballot.

This audit serves to catch any cheating by booths that might not have been detected earlier during any cut-and-choose audits. The advantage of these audits is that we are checking the candidate orders on ballot forms actually used by the voters to cast their votes rather than just on unused ballots.

We can now perform some checks of correspondence between the paper audit trail and the decrypted ballots posted from the tabulating mixes. For each selected paper ballot, the auditor now computes the adjusted seed value:

\[ \bar{s} := v + s \]

It should now be able to find a matching value amongst the decrypted outputs of the tabulation process on the WBB. Failure to find a matching value casts doubt on the conduct of the election. If the auditor finds an adjusted seed value in the tabulation that differs slightly (i.e. by less than \( n \)) from the closest seed value from the paper audit trail this may be indicative of corruption. This might be due to some manipulation of index values in the paper audit trail or the electronic records. Further investigation would now be required, firstly to establish that the paper ballot has not been manipulated.
6.3. Analysis

For ballots selected for audit for which the above check fails, we can perform a diagnostic check: we perform PET checks of the paper ballot onion against the posted receipt onions. If a match is found, and the corresponding index posted against this onion on the WBB agrees with the index of the paper copy, this would indicate that this receipt had been corrupted in the mix/tabulation phase without being detected in the audit of that phase.

We can also compute amended onions from the paper audit trail by folding the index into the LH onion in the same way that we formed the \( I \) onions. We refer to these as \( J \) onions. These \( J \) onions will have different randomisations from the corresponding \( I \) onions computed previously. However, as long as all computations have been performed correctly, the sets of \( \text{onion}_I, \text{onion}_{1_n} \) and \( \text{onion}_J \) contain the same plaintexts. In other words, The \( J \) onions should be related to the \( I \) by a series of re-encryptions and shuffles. We could test this hypothesis by performing a full PET matching of the \( I \) and \( J \) onions or, perhaps more realistically, performing some spot checks on a random selection.

6.3 Analysis

In terms of the accuracy guarantees we will see that this scheme provides stronger guarantees than Prêt à Voter 2006, assuming the integrity of the paper audit trail. If the paper audit trail is vulnerable to manipulation, then the HRPAT mechanism could undermine the assurance of accuracy of the original scheme.

Assuming the integrity of the paper audit trail for the moment, the additional auditing possibilities introduced by this HRPAT mechanism means that it will be significantly harder to violate accuracy in an undetectable way. For example, the fact that all actually voted ballot forms can be audited for correct construction means that it is essentially impossible for votes to be incorrectly encoded in receipts undetected. In previous versions of Prêt à Voter, and indeed similar schemes, these checks are probabilistic and require assumptions of lack of collusion between ballot creating processes and auditing processes.

6.3.1 Linking the receipt onions to the candidate lists

The fact that in this scheme, the ballot forms carry linked onions on both portions does create potential threats against ballot privacy. Thus, for example, if the adversary is able to link the L and R onions for a ballot form and is able to access the paper audit trail, then he will be able to compromise the secrecy of that voter’s ballot. This could be achieved with the collusion of the \( P_1 \) and \( P_2 \) processes. It is of course difficult to gauge whether this is a good trade-off, and this judgement will probably vary according to circumstance and perceived threats.

The link between LH and RH onions is cryptographically protected and cannot be directly re-established without access to a threshold set of tellers’ keys. However, there is a danger that if booth keys are compromised it may be possible to obtain the seeds for some ballots and link these to the decrypted values posted on the WBB. The coercer
still has to link the HRPAT ballot to the voter who used it. He can do this if he can establish the link between the two onions. However, our construction ensures that it would require a collusion of the $P_1$ and $P_2$ processes to reveal these links.

We see that the HRPAT mechanism does introduce some threats against ballot privacy that are absent in conventional Prêt à Voter. However, we have striven to ensure that the threshold to exploit such vulnerabilities is quite high. It is a delicate trade-off to establish whether the introduction of such vulnerabilities is justified by the added assurance and confidence resulting from the HRPAT mechanism.

### 6.3.2 Voter choices differ between pages

As the voter makes her marks on the form in the privacy of the booth, it is possible for a malicious or coerced voter to introduce different marks on the two pages in order to try to introduce inconsistencies between the paper and electronic records and so seek to discredit the election. To resolve this and to prove that the marks were made differently on each sheet by the voter the tellers can take the list of onion$_L$ and run them through a re-encryption mix to form a list of onion$_M$, as shown in Table 6.2. It is then possible to use the PET strategy to prove which onion$_M$ contains the same information as the onion$_L$, the extension of which is that the bottom page is valid but the voter’s mark does not match. If the tellers, when prompted, find that onion$_L$ with the voter’s choice $V_{\text{bottom}}$ does not have the same plaintext as onion$_R$ with the choice $V_{\text{top}}$ injected then they prove that onion$_L$ has the same plaintext as onion$_M$ to show that the marks are different on each of the pages.

### 6.4 Summary

We have presented a mechanism that can be incorporated into Prêt à Voter to generate a plaintext paper audit trail. This has a number of benefits: firstly there is the confidence building effect of having a paper audit trail as a safety-net. Secondly it provides a number of additional auditing possibilities: spot checks of correspondence between the paper ballots and decrypted ballots as well as checks on the correctness of the candidate order printed on the ballots by the booth devices. Note that these checks are applied...
directly to the candidate orders used by the voters, rather than on unused, audited forms as with the cut-and-choose audits.

A further benefit is to provide a mechanism to ensure that voters do submit the portion of the ballot that carries the candidate order, so countering dangers of voters attempting to smuggle these out to prove their vote to a coercer.

On the other hand, the HRPAT mechanism presented here does introduce some threats against ballot privacy that are not present in conventional Prêt à Voter. Evaluating this trade-off requires more systematic ways to evaluate voting systems than exist at present. Besides, it is likely that such trade-offs will be highly dependent on the context. For example, in the UK, it is required by law to maintain a link between voter id and ballot forms. Thus, in the UK, a mechanism along the lines proposed would not only be acceptable but would probably be required.

Another issue to be borne in mind, is that the paper audit trail may be vulnerable to manipulation. This is true of conventional pen and paper voting, but here it may be particularly problematic as such manipulation may serve to cast doubt on a completely valid electronic count. Again, this is a delicate trade-off against the comfort factor of having a paper audit trail fall-back.
7 Remote voting using paper-based schemes

7.1 Introduction

Postal ballots are becoming increasingly common around the world, in constituencies large and small. The motivations for this include allowing those unable to visit the polling station (such as those in the military stationed overseas; expatriates; those less abled) to cast votes from a remote location and to widen participation.

The normal procedure for postal ballots is that the materials (including ballot forms and a number of envelopes) are sent to registered voters, who have requested it, in the post. The voter is instructed to fill out the ballot form in private and to place it in an envelope. This envelope is then put in another envelope on which the identity of the voter is printed: the voter signs this envelope. This multi-layered document is subsequently posted, in yet another envelope, to the election authority.

At the election authority a number of clerks open the outermost envelopes of the received ballots. This reveals the identity of the voter who has posted the ballot and when the clerks have checked that she is eligible, the signed envelope is opened and the contents dropped into a ballot box. When all ballots have been dropped into the ballot box this is shaken so that it is infeasible for anyone to guess the identity of the posting voter. The contents of this ballot box can now be tallied together with ballots collected in polling stations.

The most obvious flaw in this process is the fact that the voter is not offered the secrecy of a voting booth. We consider a voting booth the only method of ensuring that voters are able to cast secret ballots. When sending a ballot from home, a coercer, someone in the home or an external vote buyer for example, may require to be present when the ballot form is filled out and there appears to be no way to stop this. As with traditional paper-based voting schemes the voter must also place trust in people and procedures to ensure that the election is fair and accurate.

The description of the postal ballot procedure described earlier may be a best-case scenario — many places offer, in our view, even less secrecy and security.

Some electronic voting schemes that use paper based ballot forms, such as Prêt à Voter and Punchscan, use a voting booth as a mechanism to safeguard election secrecy [82, 83]. However, it seems immediately clear that as these ballot forms do consist of paper it is possible to send them in the post, delivering the same level of secrecy (based on trust in people and processes) as traditional postal ballots.
Chapter 7. Remote voting using paper-based schemes

### 7.1.1 Properties of the scheme

The aim of this scheme is to show that it is possible to use the verification of Scantegrity together with a set of procedures to ensure that voting by post using Prêt à Voter and Punchscan can go some way toward protecting the secrecy of the vote and enabling more voters to correctly check that her vote is included in the tally. Both Prêt à Voter and Punchscan can support postal voting simply by allowing the voter to post in the encrypted receipt but because it is impossible to enforce the destruction of the secret part of the ballot form (i.e. the candidate list on the Prêt à Voter ballot form) a majority of voters would, we feel, be left with a piece of paper in their home which can, together with published election data, prove how they voted. This scheme thus aims to use a set of instructions, envelopes and procedures to destroy as much of this paper as possible at the time of voting.

**Fairness.** The proposed scheme does not impact on the fairness of the underlying voting systems as the fairness of those systems is based on the correct function of an election authority (to ensure that a proper register of eligible voters is kept). The proposed scheme also does not cause any partial information to be released before the close of the election.

**Integrity.** The proposed scheme does not impact on the integrity of the underlying voting systems but this is arguably at a very high level.

**Privacy.** The proposed scheme is intended to enhance the privacy of the underlying voting systems when they are used for postal voting. The original systems have high privacy assurances as votes are posted in to the election authority in their encrypted forms and the proposed scheme makes this as practical as possible whilst maintaining a high privacy level.

**Verifiability.** The proposed scheme adds to the underlying voting systems further verifiability which was not previously available in the postal voting setting.

**Receipt-freeness.** In the postal voting setting it is very hard to ensure that a voter does not create something that works as a receipt, perhaps by taking a picture of the filled-out ballot form before it is made into an encrypted vote and posted to the election authority. The proposed scheme can not guard against such adversarial behaviour on the part of the voter. However, a goal of the proposed scheme is to use procedures to ensure that as little “compromising” paperwork as possible remains with the voter after the time of voting. This helps remove paper that the voter should destroy but may, for any number of reasons, not have.

**Coercion-resistance.** Any voting done by post meets only low coercion resistance requirements as the system does not help voters to withstand coercion by, for example,
stopping them from voting together with someone else (by requiring the use of polling booths) and to take pictures of the ballot form. The proposed scheme cannot do anything to influence whether voters choose to vote in the company of a coerker (of any variety) but it attempts to remove as much “compromising” paperwork as possible from the majority of voters.

7.1.2 Threat model

We realise that it is impossible to stop voters from creating evidence of their vote when they vote remotely, such as a photocopy of the ballot form after they filled it in but before it is posted to the election authority. This is inherent to all remote voting schemes as it is only possible to procedurally ensure that voters are not coerced by asking them to attend a supervised polling station where they are allowed to vote in secret. The threat model here is thus the same as for all remote voting schemes: but the aim with this scheme is, among other things, to ensure that as little evidence as possible of the contents of the vote remains in the voter’s possession after she has voted. If she therefore does not create separate evidence (such as a photocopy) before she votes and she complies with the procedures described here, she will not have in her possession evidence of her vote that can be incriminating or that she, for any other reason, can present to a coerker.

Modelling the capabilities of a coerker in the postal voting setting seems almost superfluous because a postal (or online) voting system does not attempt to be coercion resistant or to help the voters to remain so. The proposed scheme gives a voter who wishes to preserve the secrecy of her vote ample opportunity to do so, but it cannot limit the capabilities of the coerker. Instead the capabilities of the coerker seem limitless: he can interact with the voter both before and during the voting takes place and is therefore able to bribe, threat or otherwise entice the voter to vote in a particular way. Even if the underlying systems (especially with this proposed scheme added on) aim to be receipt-free, a coerker who can be present at the time of voting does not need a receipt after the act of voting.

Furthermore, in the postal voting setting we cannot assume anything about the behaviour on the part of the voter: as she is not actively stopped from behaviour that threatens the secrecy of her vote she is able to do anything to comply with a coerker’s demands: by free will or under pressure. It is unrealistic to assume that a majority of voters who may be under pressure from a family member, religious leader, local party official or the like is able to find a private place to create and post the vote. All these severe privacy and secrecy threats are inherent to all postal voting and this scheme cannot influence them.

A number of further threats apply to various remote voting schemes, such as the faking of a voting credential (in the registration of a voter or perhaps the faking of a signature on a postal ballot). We here try to ensure that the (coerced) voter cannot create “evidence” of system malfunction when no such malfunction exists, but it may be possible to create faked votes — for example in order to cast doubt on the level of trust that should be vested in the system.
7.2 Techniques that do not work

It has been proposed that both Punchscan and Prêt à Voter are able to handle postal ballots simply by asking the voters to post in the encrypted receipt instead of having it scanned. However, in this section we go through a number of configurations of both systems and show why they are not as secure as immediately thought.

7.2.1 Punchscan

In the case of Punchscan the immediate technique to handle postal ballots is to allow the voter to mark her two-page form in the comfort of her own home using the required dauber\(^1\). The voter then separates the two pages, chooses one and posts this to the election authority. This page is scanned but kept secret by the authority. After the close of the election a meeting of the election officials generates not the page that the voter has posted to the authority but its twin. In short, the page published on the web bulletin board will be the same as the one the voter has still got.

It seems that this will provide the voter with a way to verify the inclusion of her vote in the tally, just as with those votes collected in polling stations. However, in polling stations the page that the voter is going to keep is scanned and immediately published onto the web bulletin board. In the postal ballot case the other page is scanned and the first is generated from this. This means that if the voter deliberately marks the two pages separately, the “receipt” will not match the page published on the web bulletin board. As there is no way of detecting whether this is because the ballot form has been maliciously altered by the authority or the voter has marked the pages differently, the page kept by the voter cannot be regarded as a useful receipt.

Furthermore, there are issues with the chain of custody of the page that has been posted in by the voter. When the authority receives the page a clerk will take it from its envelope and scan it. The clerk, who has seen this page, will be able to tell from the other page, published on the web bulletin board, how the voter has voted.

7.2.2 Prêt à Voter

It was noted quite early [20] that Prêt à Voter may be able to handle postal ballots simply by allowing the voter to fill out the normal ballot form in her home, tear along the perforation and then post the right-hand part to the election authority, but the authors were cautious as they realised this would impact on the coercion resistance of the system. Postal voting with Prêt à Voter has the advantage over traditional postal ballots that the vote is encrypted when it reaches the election authority and the receiving clerk thus has no way of discerning the voter’s choice(s). However, this does not offer the voter a receipt. Furthermore, it weakens coercion resistance even further as the destruction of the left-hand side of the ballot form, the randomised candidate list, cannot be enforced. Those voters who are able and willing to safeguard the secrecy of their vote will fill out

\(^1\)A bingo marker.
the form in secret and, as soon as possible thereafter, destroy the candidate list — but some may be unable to do so for reasons already explained.

In order to provide the voter with a receipt that she can subsequently use to check the inclusion of her vote in the tally, this scheme may be extended with a carbon paper which is overlaid the right-hand side of the ballot form. When the voter makes her marks on the ballot form these marks are thus made on two overlain sheets of papers. When she has made her marks and detached and destroyed the left-hand side of the form what remains are two copies of the encrypted receipts. The voter posts the original and keeps the carbon copy. It is quite clear that a malicious voter could mark the original and the carbon copy of the encrypted receipt separately and thus achieve the same attack as described in the previous section.

A further extension to address the destruction of the candidate list is as follows. When the voter has marked her form she detaches the carbon copy and keeps this for her own record. She then tears along the perforation and separates the two sides of the original ballot form. The left-hand side, the one carrying the names of the candidates, is placed in an envelope which has a window through which a serial number printed underneath the candidate list is visible. The voter then places this envelope together with the encrypted receipt in another envelope which has her identity printed on it. She signs this envelope and posts it to the election authority.

When such a postal ballot is received the clerk checks that the voter is eligible and registered and if so, it is checked, without opening the envelope, that the candidate list posted in by the voter is the one that matches the encrypted receipt. If this check is passed then the candidate list is shredded without being taken out of the envelope and the encrypted receipt is scanned and submitted to the web bulletin board. However, the voter might attack this solution by placing not the candidate list but another piece of paper onto which the serial number has been printed into the innermost envelope.

### 7.3 Adding Scantegrity to postal ballots

The Scantegrity system [23] proposed by David Chaum seeks to add a certain level of verifiability to all paper-based election systems, be it votes that are collected in a polling station and scanned (immediately or in a central location) or postal ballots. In essence the system is based on a chit, a corner of the ballot form, which holds the serial number of the form and is torn from it by the voter before the form is submitted. At the time of casting the voter makes a note of the randomly assigned letter that represents the choice she has made. After the close of the election the authority publishes the serial numbers of all ballot forms used together with the letters chosen by the voters. This does not leak any information about the contents of the vote but the voter can check that the published letter matches her note. If it does not, the voter can challenge the election through a procedure which proves that the chit matches the ballot form and that the correct letter has been marked on that form, without revealing the contents of the vote. See Section 2.4.4.

---

2That this happens this way can only be ensured by the use of election observers.
The Scantegrity approach immediately works with postal ballots and we simply propose the addition of the chit to both the Punchscan and Prêt à Voter ballot forms that are to be used as postal ballots. All the actions the voter must undertake in order to vote by postal ballot are thus as follows:

The voter marks her choices on the ballot form and separates it into two, by detaching the pages or the columns from each other. In the case of Punchscan she randomly selects one page to keep — her receipt. She then detaches the chit from the receipt and puts this in a safe place, along with a note of the choice(s) she has made at each position on the form. The part of the form that must be destroyed is placed into an envelope that has a window through which the serial number of this part can be read.

Along with this envelope the encrypted receipt is placed in yet another envelope which bears the voter’s identity. Having sealed this envelope the voter signs it to indicate to the election authority that she has cast her vote in accordance with the rules and so forth. With this signature she might also assure the authority that she has filled the form out in private and that she has not been asked to vote in a particular way by anyone — this, naturally, is part of election law of each particular constituency (but does not provide any formal level of assurance). This envelope is placed in an envelope which is pre-printed with the address of the receiving election authority and when this has been sealed it can be dropped in the nearest post box.

When the election authority receives the ballot, its outermost envelope is opened by a clerk who verifies the identity of the voter and her signature. If this check passes then the envelope is stripped off. The next check that must be performed is that the serial number of the part of the ballot form that has been placed in the innermost envelope matches the serial number of the encrypted receipt. This procedure is necessary to ensure that the voter returns the part of the ballot form that must be destroyed. In order to make this as secure as possible the election authority has placed a rubber stamp onto the serial number of both parts of the form. If the serial numbers match, then the clerk shreds the envelope containing the discarded part of the ballot form and scans the encrypted receipt, causing it to be posted to the web bulletin board. The original encrypted receipt is then filed according to its serial number so that it may be retrieved with relative ease at some future point. If the serial numbers do in fact not match then local legislation must determine if the vote is spoilt.

The voter is now able to visit the web bulletin board and call up her encrypted receipt, using the serial number printed on her chit. She can compare this to her notes and if they correspond she can trust that her vote is included in the tally.

In the case where the voter finds that the online representation of her encrypted receipt does not match the notes she made at the time of voting, she may use her chit to verify that her receipt was scanned correctly. She, or her representative (it may be necessary to limit the number of such checks as this can be done without compromising the correctness of the audit of the election) can take the chit to the election authority. Under the scrutiny of independent auditors and media coverage the election authority can retrieve the encrypted receipt from the archive and show, perhaps using a microscope or any level of forensic equipment required by the auditors, that the chit and encrypted receipt were once one piece of paper.
7.4 Summary

When it has been shown that the chit matches the encrypted receipt the auditors can check that the contents of the encrypted receipt does match its electronic representation published on the web bulletin board. In Scantegrity, which uses plain-text ballot forms, this check is rather tricky, please see [23, 22] for details. However, in the case of Prêt à Voter and Punchscan, where the receipt is encrypted, it is safe simply to show the receipt to the voter or any representative that she might have mandated.

We have thus modified the Punchscan and Prêt à Voter voting procedure by adding Scantegrity. Instead of keeping her actual encrypted receipt and checking that it is represented correctly on the web bulletin board the voter is able to challenge the election and have this checked in any case where she may realise that there may have been errors in the scanning process — or someone has maliciously changed her vote.

7.4 Summary

As electronic voting is becoming mature it is important to remember an old set-up that has allowed voters who have been unable to attend the polling station during election day to take part in the democratic process, namely the postal ballot. Arguably the greatest drawback of postal ballots is that the amount of trust that voters must place in people and processes (not to mention the postal service). Using Scantegrity in the paper based electronic voting schemes Prêt à Voter and Punchscan provides practical and safe auditability in the remote setting. We have introduced a series of instructions and procedures that can be put in place to make voting by post as safe as possible.

7.5 Discussion

As we feel that postal ballots are inherently less safe than ballots collected in a polling station, and in view of what we perceive to be an ever growing desire to use postal ballots, our starting point for this work has been to make this process as safe as possible. We believe we have achieved some progress toward this goal by forcing the destruction of the appropriate parts of the ballot form. It also seems that although the reliability of the election system rests on a number of procedures we have been able to remove some potential leaks of information which may compromise the secrecy of the election.

However, it is immediately clear that we have been unable to solve the most serious problem with postal ballots, namely the inability to protect the voter, and even the electoral system, from coercers who may have a completely different level of access to a large number of private homes than it may to voting booths that are under the scrutiny of observers.

3Polling booth based schemes such as Prêt à Voter provide the voter with a way of proving that this has taken place but this scheme does not.

4That is to say be as little error prone as possible and helping to safeguard the secrecy of as many votes as possible.
7.6 Future directions

As we are proposing a set of procedural safeguards for the postal ballots in Prêt à Voter and Punchscan, it is interesting to note that further procedural safeguards can be added in order to make the duties of the voter simpler and as reliable as possible. For example, if each part of the ballot form has an edge that is cut in a way that is different from the other part, then the envelope can be designed only to reasonably hold the required part of the form. Alternatively, each part of the form may be coloured differently and the colour of each envelope correspond to this.
8 Conclusions and future directions

In Chapter 3 An implementation of Prêt à Voter we present our work with the first implementation of the end-to-end verifiable electronic voting system Prêt à Voter. Starting with a set of requirements put forward by the University of Surrey Students’ Union we describe and explain design decisions and results from building a working version of the system. This work was undertaken primarily in order to enter VoComp, where we competed as one of four teams in Portland, Oregon, in the summer of 2007. Several requirements, such as the necessity of using Single Transferable Vote and allowing a number of concurrent polling stations, required a lot of thought and resulted in a set of important lessons.

In Chapter 4 Component based electronic voting systems we have introduced a hierarchical, component-based model for electronic voting systems. This model consists of four layers that each builds on the previous: physical, computational, election and human layer. The purpose of the model has not been to mimic other system modelling tools but to allow the capture and storage of information from a number of different fields in a structured way that enables developers to quickly and thoroughly adapt systems to various new requirements. One significant difference to established component based models of information technology systems is that we here propose that such “soft” parts of the system as applicable law is modelled as components that pose requirements on other components of the system.

In Chapter 5 Simple and secure electronic voting with Prêt à Voter we address two significant problems in paper-based verifiable electronic voting systems: the optical character recognition (OCR) of the marks made by voters on ballot forms is very difficult and has a certain error rate and the scanning of immensely large ballot forms, such as those used in countries where the number of contested races and the number of candidates in each race are very large, is incredibly hard. Prêt à Voter was designed to be used with ballot forms that are small enough that they can be scanned using a standard computer scanner and OCR remains error prone. The system we propose allows a voter to use a computer to form her vote, enabling her to use functions such as sorting of the list of candidates and searching for desired candidates to easily form her intended vote. The computer then prints a reduced Prêt à Voter receipt which contains a bar code that can easily and without error be scanned when the vote is cast, reducing the amount of paper needed to print a receipt and removing the need for OCR.

In Chapter 6 A human readable paper audit trail we extend a previous proposal to add a paper audit trail to Prêt à Voter in order to instill confidence in voters. Such a paper trail may add verifiability to other systems that are not verifiable, but it may also introduce a number of vulnerabilities. Our proposal therefore is to use the scheme in Prêt à Voter only until it has been agreed that the verification methods available in
Prêt à Voter do work. However, we extend the human readable paper audit trail with other properties that allow for more thorough audit of the correct construction of used ballot forms.

In Chapter 7 Remote voting using paper-based schemes we propose a set of techniques and procedures that enables the use of the paper-based electronic voting systems Prêt à Voter and Punchscan as postal ballots whilst limiting the extent of the threat to the secrecy of the election posed by remote voting.

8.1 Contribution

This thesis contributes to the understanding of electronic voting systems as follows:

- Our work with the implementation of Prêt à Voter provides a rare case study of transforming a theoretically well-specified end-to-end verifiable electronic voting system into a functional prototype. We have enumerated the problems that we encountered and that are inherent to the version of Prêt à Voter that has been implemented, but our understanding of implementing Single Transferable Vote and other non-single choice election systems, of the problems with Optical Character Recognition and of adapting an electronic voting system to varying legal requirements around the world has increased.

- Our hierarchical component-based model for electronic voting systems proposes that such systems are different from other information technology systems in that they do not only have requirements related to the systems’ ability to capture and tally votes but a plethora of requirements from non-technical domains such as election legislation and usability that have to be captured in the model. Even though we have shown that it is very hard (if not impossible) to separate some components in an electronic voting system and draw the full benefit of a component-based design, we provide the foundation for a new way of thinking about these systems.

- In proposing our “simple and secure” Prêt à Voter variant, we have placed focus on meeting real-world requirements (many candidates etc) that have hitherto been somewhat overlooked and in doing so we have proposed a usable, inclusive system that aspires to meeting high security and secrecy guarantees. Any system which uses a computer to form the vote, that is to say where software running on that computer can detect the choices made by the voter, has a set of secrecy concerns that Prêt à Voter has aimed to solve by using a paper ballot form. However, we believe that any practical end-to-end verifiable electronic voting system must be a mix of procedures and technology.

- The addition of a human readable paper audit trail to Prêt à Voter does, we argue, impose secrecy concerns that outweigh the added verifiability opportunity. Based on this we have extended such a paper audit trail with several unique auditing opportunities: the correct printing of ballot forms can be checked even for forms
that have been used to cast votes, something which has been impossible in previous versions of Prêt à Voter. It is valid to argue that we add complexity to the human readable paper audit trail, but the fundamental auditing such a trail gives is still there and we wish to make good use of the procedures in place to handle, and make use of, such a trail.

- Finally we have shown how to use three well-known end-to-end verifiable electronic voting systems (Prêt à Voter, Punchscan and Scantegrity) in postal ballots with the result that voters who follow the proposed voting procedure are helped to safeguard the secrecy of their own votes.

### 8.2 Future directions

We put forward the following directions for future research based on the work presented here:

1. No further testing of the first Prêt à Voter prototype is required at this point: a new version should be based on Prêt à Voter 2006, which makes it possible to design the implementation to be faster and more robust. However, great care should be taken to ensure that the system becomes more accessible and perhaps the key to this is to enable a number of concurrent “input” methods, i.e. the use of paper ballot forms that are scanned for most voters but the possibility to use an accessible computer for those unable to form the vote on paper. Care should also be taken not to build the new version such that it cannot (or cannot be easily altered to) handle a number of different election methods and elections with very large numbers of candidates. Preparing the system to be used in a national election should be the guiding principle.

2. An implementation of our “simple and secure” touch-screen Prêt à Voter variant may form the basis of an accessible and inclusive front-end to the new prototype. Similarly, the human readable paper audit trail shown in this thesis can be implemented in the new version — with the view to meet a potential call for a voter verifiable paper audit trail when Prêt à Voter is ready to be used in a national election.

3. Extending the component based model presented here with requirements, component definitions and security analysis of components may yield a model that is truly useful in analysing security and secrecy guarantees of the complete end-to-end verifiable electronic voting system.
References


References


References


References


[68] S. Popoveniuc and B. Hosp. An introduction to punchscan. In PRE-
PROCEEDINGS, pages 27 – 34. IAVoSS Workshop On Trustworthy Elections,
2006.

[69] S. Popoveniuc and D. Lundin. A simple technique for safely using punchscan
and prê t à voter in mail-in elections. In Proceedings of the First Conference on


and public-key cryptosystems. Proceedings of the twenty-first Communication of


[78] P. Y. A. Ryan. The computer ate my vote. Technical Report of University of


References


