EMC Submission No 12
Received 8 July 2015

Submission to the Inquiry into the Conduct of the 2014 Victorian State Election

Problems with E-Voting in the 2014 Victorian State Election and Recommendations for Future Elections

Roland Wen[1] Richard Buckland[1,2]

[1]`{roland,richardb}`[redacted]
School of Computer Science and Engineering
The University of New South Wales

[2]Australian Centre for Cyber Security

Executive Summary and Recommendations

This submission addresses two issues related to e-voting in Victoria:

i. Problems with the underlying design of the scaled-down prototype of the vVote e-voting system used in the 2014 Victorian State Election make it unsuitable for larger-scale general use in future elections.

ii. Lessons learned from the 2014 vVote trial and from similar trials internationally should guide the design, resourcing and development of Victorian e-voting systems in the future.

i. The Design of vVote

The underlying design of the vVote system is novel and mathematically sophisticated. The complete system (due to time and resource pressures the prototype version used in 2014 was a scaled-down and partial system that omitted a number of the steps and safeguards of the full design) is unique worldwide in attempting to provide 'end-to-end' verifiability for vision-impaired voters to verify that their votes were included unaltered in the final tally. End-to-end verifiability is one of a number of desirable theoretical properties for an e-voting system to have. However because the fundamental design focus of vVote was on achieving end-to-end verifiability, the resulting design suffers from three serious shortcomings that make it unsuitable for large-scale use in state elections:

- The system is highly complex: for voters to use, for voters to understand, for election officials to supervise and assist voters, for effective auditing, and to be implemented and maintained with confidence.
- Vote privacy is endangered. It is possible for hackers, insiders or well-resourced organisations to learn how individuals voted.
- The safety of the system vests in an over-reliance on the ability to detect failures during or after the election, rather than engineering a system to be unlikely to fail in the first place. Combined with the extreme complexity of the system this means the system has a non-negligible risk of catastrophic failure during an election.

There are clear benefits in having a voting system in which votes are verifiable. By having world-class verifiable voting academics to design and build the system, the Victorian Electoral Commission (VEC) has produced a design that, if fully implemented, would possibly be the most verifiable e-voting system currently used in the world. However a safe and effective electoral process needs more than verifiability; indeed the current Australian manual federal and state electoral processes are world class and do not provide any verifiability.

Focusing the design of vVote strongly on verifiability has led to unfortunate and dangerous trade-offs in the system produced. The three problematic trade-offs listed above and described in the following sections of this report make the vVote system inappropriate and unsafe to use in a critical role in Victorian elections.

It is important to note that these problems with the vVote system are inherent in its design and so cannot, for example, be patched by requesting software updates or fixes.

E-voting offers many potential advantages to voters and to those administering elections, but if e-voting is to play a critical role in Victorian elections it is important that a different design be used that balances verifiability against privacy, ease of use, ease of comprehension and trust by voters, and safe and reliable fault-tolerant operation.

We also note that, from an academic point of view, it is difficult to confidently quantify the degree to which the vVote system does indeed provide end-to-end verifiability in practice.

ii. Lessons Learned: Use of E-Voting in the Future

E-voting, whether using vVote or some other system, potentially offers considerable administrative and cost advantages in large-scale elections. It also has the potential to significantly speed up and make more accurate the counting process, which is very attractive for Australia's complex preferential electoral systems. However e-voting also introduces new risks in the electoral process that are potentially catastrophic. In all e-voting systems there are potential risks related to:

- **The increased complexity of electronic voting over more traditional pencil and paper voting**

  Complexity in e-voting systems can reduce meaningful participation in elections, for instance through disenfranchisement of less literate or less confident groups, causing misunderstandings of the process, or introducing more steps where the voter can make an error. It is also harder to design, build and assure the quality of complex e-voting systems, and these systems are more likely to have security vulnerabilities (as vulnerability counts correlate strongly with complexity).

- **Potentially catastrophic errors in the design and implementation of the system**

  Complex systems invariably contain numerous software bugs and errors. Despite considerable best efforts, even heavily resourced electronic systems such as stock exchanges, electronic banking systems, computer operating systems, smartphones, and spacecraft invariably contain unknown bugs and periodically suffer catastrophic failures. E-voting systems can crash on election day, can lose or scramble votes, or can operate incorrectly.

- **Centralised and systematic failures during the election**

  Even minor errors can be catastrophic because a fault in an electronic system has the potential to be systematic and centralised, whereas in manual systems faults tend to be random and localised. For example counting errors in manual elections in Australia tend to be random and largely cancel each other out. A counting error in software however can be systematic and impact in the same way each time, increasing the potential for a significant impact on the final result. Similarly in a manual election a loss or corruption of a ballot box (although extremely serious) impacts on a relatively small number of votes. In a centralised system the loss or corruption of a database can potentially impact on all the votes.

- **Cyber attack**

  In a manual election most attacks on the integrity of the election require an attacker to be physically present and consequently impact on a relatively small number of votes. In an electronic election attacks could be carried out remotely (even from overseas), and could potentially impact on all the votes in the election. Corrupting or disrupting an entire electronic election is significantly easier to carry out, cheaper, and less likely to be detected than in traditional manual elections. We believe it would not have been difficult for a well-funded organisation to have compromised and altered the outcome of the 2014 Victorian State Election, had it been conducted on the vVote system used in the trial.

These risks mean that elections using e-voting systems could be disrupted, election outcomes could potentially be changed and not detected, and electors could lose confidence and faith in the electoral process. Australian electoral commissions are generally regarded as being amongst the highest quality and are trusted by the electorate; it would be a tragedy if that trust was damaged or lost.

Therefore it is important that e-voting systems be developed to address and mitigate these risks. The way e-voting systems are designed, commissioned, built, evaluated and operated needs to be in accordance with best engineering practice for failure-critical systems. The authors of this submission and Vanessa Teague made a submission to the Commonwealth JSCEM Inquiry into the 2010 Federal Election [CORE11] recommending that any commissioned e-voting systems be required to be developed as failure-critical systems rather than being developed at a lower (commercial) level of quality, and we repeat this recommendation in this submission.

The team who developed and implemented the vVote system are to be commended on the remarkable job they have done with very limited resources and in a tight time frame. However unfortunately these constraints meant that vVote was not able to be designed or implemented as a failure-critical system, and so it is unsuitable to be used for more than the limited number of voters it supported in the 2014 State Election.

Recommendations

We recommend that:

1. The trial of vVote not be extended beyond the current scope of assisted and limited remote station voting.

2. Any future e-voting system must:

   1. have a strong focus on being simple for voters to use;
   2. have a strong focus on being simple to explain to voters and be easy to understand and feel confidence in;

3. be designed to ensure that the privacy of voters and their vote is comparable to the level currently provided by manual voting (or better);
4. be designed, commissioned, implemented, tested, analysed, reviewed, audited and deployed using accepted best engineering practices for critical national infrastructure; and
5. be properly and appropriately funded and resourced so that the system will be simple to use and operate, be of appropriate quality and security, and offer security and governance properties at least comparable to existing manual voting, and thus protect and preserve the currently very strong and publicly trusted Australian electoral process.

The remainder of this submission expands on each of the above issues.

1 Introduction

The primary design objective of vVote was to enable voters who cannot vote without assistance to nonetheless still be able to vote in such a way that they can verify that their vote is included unchanged in the tally, while at the same time preserving the privacy of their vote.

In general vVote goes some way towards meeting these objectives. However it is important that the usage scope of vVote does not expand to include a wider group of voters as the system has inherent problems that make it unsuitable for a critical role in public elections in Australia. In Section 2 below we expand on those problems that arise from the fundamental *design* of vVote. In Section 3 we give examples of the risks and problems that arise from the way vVote system was *implemented* and the resource constraints that bound the project. We also outline the key lessons learned from vVote, which should inform any future e-voting system for use in public elections.

2 Problems with the Design of vVote

The security and usability problems in vVote stem from its protocol design being adapted from the well-known Prêt à Voter voting scheme [RBH+09], which is a novel verifiable system, but which critically requires votes to be manually filled out on *paper* ballots. Translating the ideas of Prêt à Voter into an electronic voting scheme (where votes are filled out electronically) has weakened its security properties and lost much of the elegant simplicity of the original. It does not appear feasible to overcome these problems in the translation.

Two serious consequences of having to translate the ideas of Prêt à Voter into an electronic system are that it has become quite complex, and that there has been a trade-off where vote privacy has been weakened to the extent where it is now potentially possible for a hacker, organisation or the government to find out exactly how each voter has voted.

In this section we detail some of the resulting complexity and privacy problems of vVote, and also problems arising from the over-reliance on verifiability in the vVote project.

2.1 Complexity

The vVote system is the most complex e-voting system ever developed and implemented. Its complexity lies in both its physical procedures and its technical details.

The vVote verification procedures caused usability issues, and so certain verification steps were not used during the election. For example voters were not made aware of the need to verify their blank ballot papers as this was deemed to be too confusing, especially for voters requiring assistance to vote. However verifying blank ballots is important as one powerful cyber attack on vVote is to print fraudulent ballots. Bypassing this verification check meant that in practice the system was no longer able to provide end-to-end verification as intended. Also voters were not provided with any means to verify digital signatures, which is needed to check if ballot receipts are valid. (A ballot receipt is issued by the voting device for verifying the submitted vote. A fraudulent vote may evade detection by verification if the ballot receipt is invalid.)

The verification steps that were not omitted still had usability issues. For example for vVote to work, candidates need to be listed in random order on each blank ballot (different for each voter), and so a voter may experience difficulties in verifying that their vote correctly follows a candidate's how-to-vote card. A survey of voters who used vVote found that "[m]any respondents did not understand the purpose of the verification measures" and "[m]ore than half of respondents thought that the [ballot] receipt gave away the content of their vote, which is not the case" [BCS15].

In addition the vVote system introduced a number of unfamiliar and non-intuitive voting procedures that were required to compensate for inherent privacy vulnerabilities in the design. These voting procedures created potential difficulties for election officials and scrutineers in ensuring that the correct procedures were followed. Notably *blank* ballot papers contained sensitive information that would expose how voters voted. This meant that election officials and scrutineers were not permitted to observe blank ballot papers. This also meant that voters were required to destroy their blank ballot papers (a printout) before leaving the polling place. But on the other hand in order to provide verification, voters were encouraged to take their ballot receipts (a similar printout) out of the polling place, and so the procedure for ensuring that blank ballots were destroyed (rather than say ballot receipts) was error-prone.

Requiring blank ballot papers to be private also had broad, unintended consequences. For example election officials and scrutineers could not check that each voter was issued the correct ballot paper for their electorate since nobody was permitted to inspect the blank ballot printed for the voter.

The mathematical, cryptographic protocol and secure distributed software aspects of the vVote design and implementation were especially complex. In particular several pieces of new research work had to be carried out to design the highly complex distributed protocols to distribute trust. In turn this greatly increased the complexity of the software that had to be written.

However the theoretical benefits of this design complexity were not achieved, and the implementation and deployment did not achieve genuine distributed trust because the VEC had the sole control of vVote's implementation and operation. Distributed trust requires the co-operation and active participation of several trusted independent bodies with independent software and independent security vulnerabilities. At the time of designing vVote, it was known that distributed trust would be infeasible for the current iteration of vVote but the complexity was still added to the system. The independent review of the system implementation by DemTech observed that "vVote's distributed architecture adds considerable complexity but has questionable gain" [SBR14].

Achieving genuine distributed trust in the way required by the vVote distributed design would require multiple independent organisations to implement, operate and control separate election servers and infrastructure in different locations. This presents not only technical engineering complications but also significant non-technical challenges. For example this could change the nature of responsibility, accountability and funding for conducting elections because multiple organisations would be delegated responsibility for developing and operating the election servers and infrastructure, largely outside the control of the VEC.

Thus despite its high complexity, the current version of vVote is only a much simplified prototype system that does not yet provide end-to-end verifiability. A fully completed implementation of vVote would be even more complex and would require significant further work to include and evaluate the omitted procedures and components.

Considering that the difficulties associated with vVote's complexity were not able to be properly addressed for the current simplified version of vVote, it appears that it would be beyond the resources and the capabilities of the VEC (or any other electoral commission in Australia) to deliver the full version of vVote. A substantially simpler e-voting solution would likely have been able to achieve similar or even superior security, reliability, usability and performance properties in practice, as well as providing stronger transparency and assurance of these properties.

2.2 Vote Privacy Compromised

Australian voters have traditionally enjoyed strong vote privacy in pencil and paper elections. Indeed the modern secret ballot (also known as the Australian ballot) was first used worldwide in Victorian elections in 1856. However in the process of translating Prêt à Voter into the vVote e-voting system, this vote privacy has been compromised to the extent where it is now potentially possible for a hacker, organisation or the government to find out exactly how each voter has voted.

In secret ballot voting, blank ballot papers are identical and anonymous. Provided that nobody observes a voter filling out their ballot paper, once the completed ballot is placed in the ballot box an attacker cannot distinguish it from other ballots in the box to learn how that voter voted. In such elections a vote is private by default and nothing additional needs to be done to ensure a voter's privacy. In vVote however the blank ballot papers are all different and are marked with a serial number. In order to keep a voter's vote private a series of physical, cryptographic and secure software systems all need to operate correctly and not fail or be compromised before, during or after the election.

Consequently the vVote system introduces a range of new vote privacy vulnerabilities to the electoral process. These vulnerabilities can potentially be exploited to coerce voters or even to reveal how all voters voted. We have already discovered one attack on vVote that exploits one such privacy vulnerability (described in detail in Section 2.2.1 below). In traditional manual secret ballot voting and other e-voting systems used in public elections (including the VEC's previous e-voting system), the same vulnerabilities are either less serious or not present.

Many of the privacy vulnerabilities are inherent in the vVote design. In particular the design requires a voter's blank ballot paper to remain hidden from everyone else even before it is filled out by the voter, and also after the voter has voted, as anyone who observes the blank ballot can later deduce the voter's vote. Depending on blank ballot papers being kept secret before and after the election is a radical weakening of traditional secret ballot voting, and is the source of a number of privacy vulnerabilities. For example vulnerabilities are introduced by the practical difficulties in enforcing procedures to prevent the observation of blank ballots by election officials and to ensure their destruction after voting. Also ballot printing devices and software systems become a target for privacy attacks because these devices contain sensitive information that exposes the votes (unlike in other e-voting systems, where typically only the voting devices contain such sensitive information).

Furthermore the impact of a successful vote privacy attack is more serious because votes may be tied to voter identities through metadata. The VEC successfully trialled an electronic roll mark-off system at early voting centres in 2010, and plans to expand its use. The electronic roll mark-off system generates a wealth of metadata, including timestamps of when voters were marked off the roll. E-voting systems also generate metadata, including timestamps of when votes were cast. The voter mark-off timestamps and the vote casting timestamps can be correlated to reveal probabilistic information about how a voter has voted and potentially to directly link voters and their votes, thereby violating vote anonymity.

For typical e-voting systems, this metadata link between voters and their votes may not be strong — there is a gap between the time that a voter is marked off the roll and the time that the voter casts their vote, and so uncertainty is created if several voters are voting around the same time.

However the vVote system generates additional metadata that can be used to create a strong link between voters and their votes. For example the ballot printing devices and the central election servers generate metadata including timestamps of when ballot papers were issued (and ballot papers are explicitly linked to votes). Since there is a negligible gap between the time that a voter is marked off the roll and the time that the voter is issued a ballot paper, it is quite likely that voter mark-off timestamps and the ballot issuing timestamps provide a strong link between voters and their votes.

Consequently a successful privacy attack could potentially reveal how all vVote voters voted. While this is a known vulnerability in unsupervised Internet voting, which is not secret ballot voting, this is not expected in supervised e-voting in a polling place, which should be secret ballot voting. It would seem relatively straightforward for an insider to carry out such an attack during or after the election, or for a cyber adversary to carry out a man-in-the-middle attack on the polling place network traffic.

The full extent of the vVote privacy vulnerabilities and the risks they pose remain poorly understood as they have yet to be studied in detail by security experts and election administrators. Even where some of these vulnerabilities are known, the countermeasures in vVote are weak or even flawed. For example the attack we describe below exploits a flawed countermeasure against a known vulnerability in vVote.

Although it is possible that as individual privacy vulnerabilities from this dangerous design are disclosed they might be able to be patched, this is not an appropriate approach for creating a failure-critical system. Security needs to be designed into the system from the outset. Security engineering experience has shown that patching problems as they arise does not turn an insecure system into a secure system. Patches generally further increase complexity of already complex systems, and frequently are at least partially ineffective or introduce other vulnerabilities. Critically it may be that there are vulnerabilities that cannot be patched, and/or undisclosed vulnerabilities that are known to attackers but not to the system designers.

We now give an example of an attack that exposes privacy vulnerabilities in vVote. It is important to note that the fundamental problem is not the details of this one particular attack, but rather that such attacks are made possible in the first place due to the complexity of the system and the inherent danger of having ballots with serial numbers on them.

2.2.1 Example Attack on Vote Privacy

The vVote system currently has a vote privacy vulnerability that allows a malicious party to carry out a 'chain voting' attack. This vulnerability results from a flawed assumption in the vVote security model and also from the inherent privacy weakness of the ballot design. Using chain voting an attacker can violate vote privacy and use this to coerce a series of voters to vote as directed (for instance via vote buying or intimidation). Interestingly the vVote design does have a countermeasure specifically included to prevent chain voting, but this countermeasure is flawed and easily circumvented.

First we give a brief overview of the fundamental vote privacy weakness in the vVote design. Then we describe chain voting and vVote's countermeasure against this attack. Finally we describe the vulnerability in this countermeasure, and a simple way an attacker could exploit the vulnerability.

Vote Privacy Vulnerability To cast their vote in vVote, a voter is first issued a uniquely marked ballot paper and then scans this ballot paper into a voting device. The voter uses the voting device to display their vote and then to submit their vote electronically. The voting device then prints a ballot receipt for verification (this ballot receipt does not reveal the vote by itself). The original ballot paper remains blank and is not further marked in any way.

The vVote design has two key features that create an inherent privacy vulnerability.

1. The ballot papers are designed to help provide certain integrity and privacy properties. This design requires each blank ballot paper to contain particular information that is unique to that ballot paper[1].

2. The ballot receipts are openly published for verifiability purposes, and so anyone can obtain all the information in all the ballot receipts[2].

The inherent privacy vulnerability is that combining a (blank) ballot paper with its published ballot receipt will reveal the corresponding vote that was cast[3]. Thus a voter could deliberately violate their own vote privacy by revealing their (blank) ballot paper. This ability to produce evidence of a vote makes it possible for an attacker to coerce voters. ("Vote for Fred and show me your blank ballot to demonstrate that you did — or I'll break your leg / and I'll give you a lottery ticket.")

This vulnerability is a well-known problem, and so the vVote system includes two physical countermeasures to help protect privacy:

1. Nobody is permitted to inspect a blank ballot paper except for the voter.

2. Each voter must destroy their blank ballot paper before leaving the polling place.

In practice however it is difficult to ensure that what a voter destroys is their ballot paper because election officials are not permitted to view ballot papers. The vVote design therefore anticipates the likely possibility that some voters will be able to take their ballot paper out of a polling place, and so includes a countermeasure to prevent a potential chain voting attack, which such an action could allow.

Chain Voting and Countermeasure Chain voting is a simple and low-cost attack that permits coercion of multiple voters and arises in traditional voting systems when an attacker is able to obtain a single, unused (blank) ballot paper from a polling place. The first coerced voter completes this ballot outside the polling place under the scrutiny of the attacker, and then goes into the polling place, collects a new blank ballot (which they later smuggle out), and casts instead the previously completed ballot. This can then be repeated indefinitely over a 'chain' of coerced voters.

The chain voting attack could be adapted to vVote in the following way.

After obtaining an unused ballot paper, the attacker gives this unused ballot to the first coerced voter in the 'chain', along with instructions on what vote to cast. Then the attack involves the following steps.

1. The coerced voter enters the polling place and is issued a new ballot paper by an election official. But instead of using this new ballot paper to vote, the voter uses the ballot paper provided by the attacker. The voter then destroys the attacker's ballot paper, to give the appearance of complying with the voting procedures.

[1]Each ballot paper contains a serial number and a randomly ordered list of the candidates.

[2]Each ballot receipt contains the ballot serial number and the order of the preference rankings for the candidates, but not the (random) order of the candidates.

[3]A vote is revealed by learning both the order of the candidates (from the ballot paper) and the order of the preference rankings (from the ballot receipt).

2. The coerced voter takes the unused ballot paper (issued by the election official) and gives it to the attacker.

3. The attacker repeats the attack by giving this unused ballot to the next voter in the ‘chain’.

Note that the attacker can easily check whether each coerced voter has voted as instructed. This is because of vVote’s privacy vulnerability, as described above — the attacker learns a coerced vote by inspecting the corresponding ballot paper and openly published ballot receipt.

The vVote design includes a countermeasure against chain voting: a ballot paper will expire five minutes after being issued to a voter if the ballot has not yet been used to start voting. The premise behind this countermeasure is that the short time limit for each ballot to be used to start voting would make chain voting impractical.

Vulnerability in vVote Countermeasure The chain voting countermeasure has a vulnerability that makes it possible to circumvent the time limit: a flawed assumption is made that chain voting is possible only if the attacker can inspect a ballot paper *before* it is used by a coerced voter to vote.[4] In fact the attack is still possible if the attacker can inspect the ballot paper *later*.

Exploiting this vulnerability requires a slight modification to the original chain voting attack. The modification is a partial reversion to the regular voting procedure — now a coerced voter votes using the ballot paper issued by an election official, instead of using the ballot paper provided by the attacker. However the voter still destroys the attacker’s ballot, and the rest of the attack also remains the same.

The steps for the modified chain voting attack are as follows.

1. The coerced voter enters the polling place and is issued a new ballot paper by an election official. The voter immediately uses this new ballot paper to vote (thereby nullifying the time limit countermeasure). The voter then destroys the attacker’s ballot paper, to give the appearance of complying with the voting procedures.

2. The coerced voter takes the ballot paper (issued by the election official) and then gives this ballot to the attacker (who thus learns the vote).

3. The attacker repeats the attack by giving this ballot to the next voter in the ‘chain’.

It seems difficult to mitigate this modified chain voting attack because the vulnerability stems from a privacy vulnerability inherent in the vVote design.

2.3 Over-Reliance on Verifiability

The safety of the vVote system vests largely in the potential ability to detect failures during or after the election, rather than in having engineered the system to be unlikely

[4]This assumption is correct for chain voting attacks on traditional secret ballot voting. The flaw appears to be caused by applying this same assumption to vVote.

to fail in the first place. Combined with the extreme complexity of the system this means the system has a non-negligible risk of catastrophic failure during an election. This failure may well be *detected* by the verification processes but that is not as useful as having been able to *prevent* the failure.

Verifiability can be likened to an aircraft black box flight recorder. It can be very useful after a disaster to help identify what went wrong for next time, but most passengers would probably prefer to fly in aircraft designed and engineered not to crash, rather than one which provides useful diagnostics afterwards.

The vVote system was designed from the outset to be verifiable, and the design was principally driven by this single property; the design team consisted largely of world experts in verifiability. However this came at the cost of other important properties, including privacy, integrity, reliability, usability and transparency.

Fundamental engineering and oversight practices for failure-critical systems were not followed throughout the vVote project, on the basis that these were unnecessary for a verifiable system. It was argued that a system does not need to follow processes designed to minimise the chance of failing if failure or lack of failure can be proved after each election. Instead of adopting highly rigorous levels of engineering and project management, as needed to ensure the quality of such critical systems, key engineering and project management activities were reduced or completely bypassed. This approach resulted in engineering weaknesses and a lack of assurance that system failures would be prevented.

Rigorous software engineering practices were not adopted. For example the implementation was carried out by three separate organisations, each of which had a single person implementing most of the code. This approach precludes many software engineering best practices for reliability (and also some simple coding basic practices), which depend on team collaboration and multiple eyes continuously reviewing during implementation, rather than a single person. The DemTech Review [SBR14] found that "the quality and maturity of the different parts of the system varies considerably", and identified numerous software design and implementation problems (for instance a 20-page appendix was devoted to examples of code issues), many of which could not be fixed before the election. These problems included highly critical issues such as brittle fault tolerance, missing failure recovery functionality and weak randomness, as well as basic software engineering code quality issues such as:

- missing comments and comments that are inconsistent with code,
- poor readability, maintainability and encapsulation,
- poor naming (resulting in errors in several instances),
- hardcoded values,
- inconsistent array indexing (which was "worrisome for production code"),
- excessive code duplication, and

- redundant code.

It is utterly inappropriate for critical national infrastructure to be developed with this level of quality and engineering control. The particular problems identified are not the most alarming aspect of the code review — rather it is that these problems were able to creep into the system and that they are so basic as to indicate that proper software engineering was not followed.

The second author of this submission has trained over 10,000 of Australia's top software engineers over two decades and would expect higher quality code and coding practices from even first year engineering students.

The vVote implementation also made extensive use of third-party software of unknown quality. Such external software can introduce security and reliability vulnerabilities, as was the case for the NSW iVote system, because third-party software is "often not engineered to the level of security that is required for critical, high-risk applications. ... Given the economic and foreign policy stakes involved in the outcome of a large election, such contests need to be treated as national security matters, which require a wholly different technical approach than typical IT systems" [HT15].

There were numerous other engineering shortcomings. The vVote system did not have detailed requirements specifications to define the system's properties. A number of the most critical components (such as the mix-net and cryptographic functions) had poorly defined requirements. The documents describing the requirements and design were incomplete and inconsistent, and contained requirements that were violated by the design and implementation, as well as requirements that were later acknowledged as "unattainable" [SBR14]. The VEC conceded that "[i]t emerged during development that there were details that were missing or contradictory" [VEC14].

Many of the requirements, design and implementation problems may be largely attributed to the failure to put the software implementation phase of vVote to competitive tender, to find suitably qualified developers. Instead members of the vVote design team themselves were contracted to implement the bulk of the system despite their lack of resources, software engineering experience and large-scale critical infrastructure expertise.

The tight coupling of design and implementation was contrary to the usual engineering principle of strong separation between design and build responsibilities. For example as a byproduct of its review of large-scale software engineering projects, the NSW Independent Commission Against Corruption (ICAC) identified strong separation of design and build as one of the five key elements to ensuring the quality and success of IT projects [ICAC13].

The risk of these engineering and software management problems occurring and introducing faults and vulnerabilities into the system was not well understood and anticipated due to shortcomings in the processes for assessing and managing risk. In addition scheduled independent audits were not performed on the vVote system. There was no pre-election audit, which could have helped to inform the final decision to use vVote in the election by identifying risks and problems in the system itself and also the engineering and project practices. Likewise there was no post-election audit to report on the problems and failures that occurred during the election and that need to be addressed in the future. Carrying out such independent audits and then publishing reports is also important for facilitating public transparency and scrutiny, as was the case for the NSW iVote system in 2011. (The PWC audit reports for iVote [PWC11b; PWC11a] were valuable sources of information for our submission to the NSW JSCEM Inquiry into the Administration of the 2011 NSW State Election [CORE12].)

The reasoning given for not performing audits was that they were obviated by vVote's verifiability property [VEC14]. But in reality the high complexity of vVote and its associated non-intuitive, highly mathematical verifiability features demanded even greater rigour and scrutiny than for other failure-critical systems.

This over-reliance on verifiability may have been in part due to false confidence in the ability of vVote's verifiability property to detect errors in the election outcome. For example a flawed statistical analysis of the vVote verification process claimed that a high level of confidence in the election outcome could be achieved with only a relatively small number of verification checks being performed [CRST14]. However that analysis was invalid as it implicitly assumed that the votes verified by voters would be a random sample of the votes; this assumption is clearly not valid in practice. It is questionable if any quantitative conclusions can be drawn from measurements of the verification checks performed, and if the measurements themselves are meaningful at all.

Thus while verifiability is a useful property, it has its limitations. Verifiability can help to *detect* some integrity failures but cannot *prevent* them. For e-voting systems, verifiability should be considered as a 'last line of defence'. Extensive rigour and effort in a wide range of engineering and project management areas is necessary to provide multiple layers of defence to prevent integrity failures, as well as to prevent failures in all critical properties, including privacy, reliability and even verifiability itself.

3 Lessons Learned: Use of E-Voting in the Future

The problems of and experience with the vVote system provide valuable lessons for future e-voting systems in Victoria and Australia. Many of these lessons relate to well-known risks in developing e-voting systems that have been discussed extensively, for instance in a submission to the Commonwealth JSCEM Inquiry into the 2010 Federal Election [CORE11]. Despite the best of stated intentions of the vVote project, resourcing and scheduling constraints meant that these known risks were not addressed and in some instances were exacerbated. For e-voting systems designed and used in the future, it is important that the development and implementation risks are assessed and managed according to risk practices appropriate for failure-critical systems. One of the important lessons of vVote is that nation critical software systems need to be resourced appropriately for the desired level of system quality; otherwise project pressures will force unanticipated and potentially risky compromises to be made.

Below we give some examples of lessons on e-voting risks that were not addressed by vVote, and the lessons learned. For the interested reader a more extensive and detailed discussion of the vVote risks and lessons can be found in a separate technical report [WB15].

3.1 Risks of Increased Complexity of Electronic Voting over More Traditional Pencil and Paper Voting

Manual paper-based voting processes are generally well-understood and carefully scrutinised. In Australia this can be largely attributed to the processes being intuitive and simple, and to the longstanding and ongoing efforts of electoral commissions in facilitating the public understanding of these processes. These are necessary for meaningful participation and scrutiny, and thus a high quality electoral process.

However electronic voting processes are inherently opaque, and can feel non-intuitive and less trustworthy to voters due to their complexity. The vVote system used was far more complex than originally anticipated, and this created risks for those using and those supervising the system, as well as those developing and assuring the quality of the system.

For example surveys of election officials supervising vVote revealed that "more than half of respondents stated the system was Too Difficult to Operate or Not Very Reliable" [BCS15] and that, alarmingly, election officials did not understand verifiability. Without being able to understand the system and have confidence in its reliability, it seems likely that election officials would have experienced difficulties in ensuring that voters correctly followed procedures and in properly assisting voters who encountered problems using vVote. Moreover if election officials themselves did not understand the system, the properties it possessed and the reasoning behind some of the more non-intuitive steps, it is more difficult for the voters to have confidence in the system.

The design was the primary source of complexity in vVote. The extensive research work involved meant that the design took a year longer than expected to develop, and in turn the ensuing high design complexity meant that the implementation also took a year longer than expected to develop. These delays were well outside the maximum tolerance limits in the original project schedule [VEC11].

The high complexity of vVote's design and implementation increased the risks that vVote's quality and maturity could not be assured in time for the election. Unfortunately the severe schedule slippage cut the time available for quality assurance activities to less than half that originally scheduled [VEC11; BW12; WB15], and this resulted in many quality assurance activities being compressed and/or omitted. Yet at the same time the higher-than-expected complexity of vVote meant that greater rigour, time and resources for quality assurance was needed, rather than less.

Furthermore a consequence of vVote not being developed in accordance with rigorous engineering practices for failure-critical systems was that a more protracted quality assurance process was needed to patch or attempt to mitigate the defects and vulnerabilities that were discovered, and then to reassess the system.

While some degree of complexity is unavoidable in e-voting systems, the risks arising from complexity are best mitigated by vigorously minimising the level of complexity as a key design goal. The risks of any residual unavoidable complexity must then be carefully managed by ensuring that rigorous quality engineering processes are planned and followed for designing and implementing e-voting systems, and assuring their quality.

3.2 Risks of Potentially Catastrophic Flaws in the Design and Implementation of the System

Failure-critical systems can have catastrophic failures arising from very minor design and implementation defects. For example the investigation into the Ariane 5 Flight 501 disaster, one of the most costly software failures in history, traced the disaster to a single line of code. The board of investigation found that an inadequate review process was a major contributing factor, that integrating multiple levels of review into the development process is “the overriding means of preventing failures”, and that “software should be assumed to be faulty until applying the currently accepted best practice methods can demonstrate that it is correct” [AIB96].

However a rigorous review process was not incorporated into the vVote project to manage the risk of catastrophic flaws. As a result key areas such as the requirements, design and practices did not undergo comprehensive engineering reviews. Only a single, tightly constrained engineering review was carried out, by DemTech, close to the end of the project.

The DemTech Review [SBR14] had a very narrow scope (it was principally to check for consistency between the specifications and the implementation) and took place over a brief period (only two weeks). The review was not conducted until the implementation was almost complete, and this was too late to address the numerous problems discovered in time for the election. A continuous review process would have identified many of these problems (and others) early on and led to significant improvements. The DemTech Review cautioned that “[e]ven minute design or implementation flaws can result in unexpected system failures. While this is true for most software systems, we emphasize that the complexity of the algorithms used [in vVote] is quite high and the combination of cryptography and distributed computation leads to extremely subtle interactions”.

Instead of engineering and project reviews, it appears that the quality assurance of the vVote project was heavily reliant on theoretical academic peer review of papers derived from the project and authored by the architects of the system. This is reflected in the emphasis on academic papers over engineering specifications in the project documentation, for instance the DemTech Review found that “in the existing documentation, essential requirements and design aspects are not given in the 6 project documents, but rather in supplementary material like [the research paper ‘Draft technical report for VEC vVote System’]” [SBR14].

Academic peer review is not a substitute for proper engineering review. Although academic peer review certainly has value, it is quite different from and is not a substitute for engineering and project reviews. Peer reviews of academic papers do not have an investigatory role — peer reviews typically accept statements and claims made without any attempt at verification and instead focus on checking reasoning and conclusions. Furthermore the paper acceptance process is not an independent and arms-length review of the matters addressed in the paper — instead the authors of an academic paper select and provide the raw data or a summary of the raw data, and propose their own conclusions. Reviewers have varying and essentially unknowable levels of expertise and interest in the papers they review, and devote an uncontrollable amount of time to the review but typically only in the order of magnitude of a few hours. The timing (feedback is delayed and ongoing feedback is not possible) and scope (which cannot be set) of academic peer reviews are poorly suited to an ongoing engineering project.

The role of academic peer review of e-voting systems is not to provide an in-depth analysis that identifies all problems and vulnerabilities, but rather to facilitate such analysis by disseminating knowledge of the system through publication. Only then does the broader analysis and review begin, and this is a lengthy process. For many e-voting systems and cryptographic protocols in the literature, serious flaws and vulnerabilities have been discovered years after the papers were published.

The vote privacy attack on vVote that we described in Section 2.2.1 is an example of how easily flaws and vulnerabilities can be overlooked by those designing and implementing an e-voting system, and those who analyse, review and test the system. In this case the faulty chain voting countermeasure was published in a peer reviewed paper despite having an obvious flaw. Given that we discovered the attack through a highly ad hoc, superficial inspection of vVote, it seems clear that a systematic, comprehensive, in-depth analysis and review process could reveal even more serious flaws. Thus a rigorous review process is needed to add layers of defence against catastrophic flaws in e-voting systems.

3.3 Risks of Cyber Attack

Given the high stakes of elections, e-voting systems are susceptible to a wide range of cyber attacks by a diverse range of attackers with different motivations. Examples of the more serious attacks on an e-voting system include changing the election outcome by compromising election integrity, disrupting the electoral process by denial of service attacks, and corruptly influencing voter behaviour after compromising voter privacy (this influence may even extend to the behaviour of compromised voters outside the electoral sphere).

E-voting systems need to satisfy many security properties to mitigate the risks of such attacks. But although e-voting systems used in elections are claimed to satisfy important security properties, in many cases they do not. As a result the risk of cyber attacks is not clear, and so there is often false confidence in e-voting systems.

Verifiability is especially problematic as there are many complex aspects to genuine verifiability. E-voting systems used in public elections have been claimed to be verifiable when in fact the system has flaws in its verification processes, or is not verifiable at all. For example the e-voting system used in the 2010 Victorian State Election had a mechanism for voters to “verify” that their votes were successfully submitted and processed, but in reality this was easy to circumvent and did not provide any meaningful verifiability.

The vVote system used in the 2014 Victorian State Election was developed with the intention of addressing this failure, but was still not genuinely verifiable as there were significant gaps in what could be verified. Moreover even if these gaps were filled, it could still be unclear if vVote is indeed a genuinely end-to-end verifiable e-voting system. Its verifiability relies on abstract mathematical arguments, which are difficult to check especially given the high complexity of vVote.

Some of the verifiability gaps in vVote in the 2014 election arose from features of the vVote system that were not able to be used in the election because they were either too complex and confusing (for instance voters were not told to verify blank ballots), or not completed (for instance voters could not verify the validity of their ballot receipts as there were no means provided to verify digital signatures in the polling place).

Verifiability gaps also arose because vVote did not enable *independent* verification. The crux of verifiability is that anyone who wishes to do so can independently verify all aspects of the system. Verifiable systems may give confidence that the correct result is obtained but, ironically, only if the verification software itself is correct, secure and bug free. "Who verifies the verifier?" is an important question, particularly in the case of vVote where the verifier is potentially more complex than most entire e-voting systems.

Having assurance in the quality of the verifier is only possible if the system provides strong transparency, and a key part of strong transparency is accurate, comprehensive engineering specifications. However the vVote system did not have detailed engineering specifications, and this made two core verification tasks infeasible.

First it was infeasible to independently perform a detailed analysis of the design, implementation and deployment of vVote to check for flaws in the highly complex verification processes. The extensive, broad, public review needed to challenge the veracity of vVote's verifiability claims was not possible. The DemTech Review of vVote raised concerns about these documentation problems: "Although documentation may appear to be a secondary concern, we nevertheless strongly recommend improvements here as it directly impacts implementation, quality assurance, and maintenance. In its current state, **we found it difficult, and in some cases, impossible, to match parts of the implementation with the received design documents** [emphasis added]" [SBR14]. (Note that DemTech also had access to material that is not publicly available.)

Second it was infeasible to independently implement verifier software to check for flaws in vVote's verifiable artefacts as originally intended. The VEC provided its own verifier software for checking certain public artefacts but this was not independently implemented, was based on unpublished system details, and may itself have been flawed. Also certain private artefacts were not able to be independently verified. For example vision-impaired voters were only permitted to verify their votes using devices and software provided by the VEC.

Thus despite the substantial efforts dedicated to developing a genuinely verifiable system, the vVote system was not able to provide verifiability in practice, and so vVote was not able to counter attacks on integrity in the ways intended. Again this was largely due to vVote's complexity — the more complex a system's procedures and technical details are, the more difficult it is to make verifiability usable and transparent, and thereby genuine and meaningful.

Furthermore the heavy focus on verifiability appeared to divert efforts from ensuring that vVote mitigated the risks of attacks in other areas. In fact the way that vVote is designed to provide verifiability significantly increases an election's attack surface for numerous attacks (including on integrity), compared to both manual systems and the VEC's previous e-voting system. There remains a great deal of uncertainty over what attacks are mitigated or exacerbated by vVote, and for future e-voting systems the risks must be made explicit and public well in advance of an election.

References

[AIB96] Ariane 5 Inquiry Board. *Ariane 5: Flight 501 Failure*. European Space Agency, 1996.
URL: http://esamultimedia.esa.int/docs/esa-x-1819eng.pdf.

[BCS15] Craig Burton, Chris Culnane and Steve Schneider. "Secure and Verifiable Electronic Voting in Practice: the use of vVote in the Victorian State Election". In: *CoRR* abs/1504.07098 (2015).
URL: http://arxiv.org/abs/1504.07098v1.

[BW12] Richard Buckland and Roland Wen. "The Future of E-voting in Australia". In: *IEEE Security & Privacy* 10.5 (2012), pp. 25–32.
URL: http://dx.doi.org/10.1109/MSP.2012.59.

[CORE11] Roland Wen, Vanessa Teague and Richard Buckland. *Best Practices for E-election Systems. Computing Research and Education Association of Australasia (CORE) Supplementary Submission to the Inquiry into the 2010 Federal Election.* Submission 101.1, Inquiry into the 2010 Federal Election. Joint Standing Committee on Electoral Matters, Parliament of Australia, 2011.
URL: https://www.aph.gov.au/Parliamentary_Business/Committees/House_of_Representatives_Committees?url=em/elect10/subs/sub101.1.pdf.

[CORE12] Vanessa Teague and Roland Wen. *Problems with the iVote Internet Voting System. Computing Research and Education Association of Australasia (CORE) Submission to the Inquiry into the Administration of the 2011 NSW Election and Related Matters.* Submission 7, Inquiry into the Administration of the 2011 NSW election and related matters. Joint Standing Committee on Electoral Matters, Parliament of NSW, 2012.
URL: http://www.parliament.nsw.gov.au/prod/parlment/committee.nsf/0/BA09355EDE5E3859CA2579AD0001D53C.

[CRST14] Chris Culnane, Peter Y. A. Ryan, Steve Schneider and Vanessa Teague. "vVote: a Verifiable Voting System". In: *CoRR* abs/1404.6822 (2014).
URL: http://arxiv.org/abs/1404.6822v3.

[HT15] J. Alex Halderman and Vanessa Teague. "The New South Wales iVote System: Security Failures and Verification Flaws in a Live Online Election". In: *CoRR* abs/1504.05646v2 (2015).
URL: http://arxiv.org/abs/1504.05646v2.

[ICAC13] NSW Independent Commission Against Corruption. *Managing IT contractors, improving IT outcomes*. Discussion paper. 2013.
URL: http://www.icac.nsw.gov.au/component/docman/doc_download/4191-managing-it-contractors-improving-it-outcomes-august-2013.

[PWC11a] PricewaterhouseCoopers. *Technology Assisted Voting Audit: iVote Post Implementation Report*. 2011.
URL: http://www.elections.nsw.gov.au/__data/assets/pdf_file/0007/93481/iVote_Audit_report_PIR_Final.pdf.

[PWC11b] PricewaterhouseCoopers. *Technology Assisted Voting Audit: iVote Pre Implementation Report*. 2011.
URL: http://www.elections.nsw.gov.au/__data/assets/pdf_file/0006/93480/iVote_Audit_report_pre_imp_7_March.pdf.

[RBH+09] Peter Y. A. Ryan, David Bismark, James Heather, Steve Schneider and Zhe Xia. "Prêt à Voter: a Voter-Verifiable Voting System". In: *IEEE Transactions on Information Forensics and Security* 4.4 (2009), pp. 662–673.
URL: http://dx.doi.org/10.1109/TIFS.2009.2033233.

[SBR14] Carsten Schürmann, David Basin and Lorena Ronquillo. *Review of the vVote System*. DemTech Group, 2014.
URL: http://vec.vic.gov.au/files/demtech.pdf.

[VEC11] Victorian Electoral Commission. *vVote Project Plan 1.0*. 14th Nov. 2011.

[VEC14] Victorian Electoral Commission. *Victorian Electoral Commission comment on DemTech Report*. 2014.
URL: http://vec.vic.gov.au/files/demtech.pdf.

[WB15] Roland Wen and Richard Buckland. *Lessons Learned for E-Voting: A Case Study of the VEC vVote Project*. Tech. rep. UNSW-CSE-TR-201509. (To appear). UNSW Computer Science and Engineering, 2015.
URL: https://www.cse.unsw.edu.au/~reports/.