Submission
No 17

ADMINISTRATION OF THE 2019 NSW STATE ELECTION

Name: Dr Vanessa Teague

Date Received: 9 October 2019

Internet voting and election verification

Dr Chris Culnane[*]and A/Prof Vanessa Teague[†]
The University of Melbourne
Prof Rajeev Goré[‡]
The Australian National University

October 9, 2019

This submission contains similar material to our previous submissions to earlier instances of this committee. The main point is unchanged: there is no current method of voting remotely over the Internet that protects privacy and defends the integrity of the election result adequately for government elections. There are a number of reasonable solutions for e-voting via a computer in a polling place—suggestions are given below.

Recommendations

Most of these are the same as those in our previous submissions.

Recommendation 1 Discontinue internet voting.

Recommendation 2 Ensure that any pollsite electronic voting system has either a voter-verified paper record or a genuine form of end-to-end verification.

Recommendation 3 Conduct a statistical audit of the Legislative Council paper ballots after counting.

This submission incorporates work and analysis by Neal McBurnett, Mark Eldridge, Aleks Essex, Rich Garella, Alex Halderman, Joe Hall, Sarah Jamie Lewis and Olivier Pereira.

We would be happy to discuss any of these issues further with the committee.

[*] Chris Culnane is a Lecturer in the School of Computing and Information Systems at the University of Melbourne, with research interests across cyber security and privacy. From 2012 to 2015 he was the Technical Lead for the University of Surrey on the vVote project run by the Victorian Electoral Commission (VEC) to develop an end-to-end Verifiable Election System. The system was deployed as the Electronically Assisted Voting solution for the 2014 State election.

[†] Vanessa Teague is an Associate Professor in the School of Computing and Information Systems at the University of Melbourne. She is the chair of the Cybersecurity and Democracy Network, an advisory board member of Verified Voting, and was a contributor to Victoria's vVote e-voting project.

[‡] Rajeev Goré is a Professor and Associate Director of Research for the Research School of Computer Science at the Australian National University. His research interests are in Electronic Voting and Vote-Counting, Proof Methods for Non-classical Logics, Term Rewriting, Interactive Theorem Proving, Automated Reasoning and Logic.

1 Introduction

Apart from the obvious privacy implications, the key concept in electronic elections is *verifiability*, the opportunity to check whether an announced election outcome is accurate. Plain paper voting in a polling place can be verified by observers and scrutineers; computerised voting is much harder to observe, because watching the screen gives scrutineers no real evidence of what the computer is doing with the votes.

Anyone can claim that their system is secure and protects people's privacy, but elections must demonstrate that the result accurately reflects the choice of the people, to the satisfaction of scrutineers, disappointed candidates, and members of the public.

iVote has proven nothing about the safety and security of Internet voting. The best that can be said for it is that it has not yet been successfully challenged in court by a disappointed candidate who doesn't accept their electoral defeat. This is not the same as saying it has proven its safety—it's similar to saying that surviving a drive down the freeway without a seatbelt proves that seatbelts are unnecessary. Election evidence is a safety feature that becomes necessary only in the case of close elections, contested elections, or the erosion of public trust in elections. iVote hasn't yet experienced any of these danger scenarios, but one day it will, and at that point it will become obvious that its results do not prove anything about the election outcome.

iVote elections are not verifiable. The votes might have been manipulated or accidentally altered, or they might not, but there is no way to verify their accuracy.

2 iVote's history of security issues

iVote has been proven vulnerable to fraud as a result of a series of serious errors and security problems. In 2015, our team found that the iVote site was vulnerable to an internet-based attacker who could read and manipulate votes [HT15]. The attack would not have raised any security warnings at either the voter's or the NSW Electoral Commission (NSWEC) end, but it should have been apparent from iVote's telephone-based verification. When the NSWEC claimed that "some 1.7 per cent of electors who voted using iVote also used the verification service and none of them identified any anomalies with their vote,"[1] we took that as reasonable evidence that the security problem had not been exploited. But this claim was not true. A year later it was revealed that 10 per cent of calls to the verification service hadn't been able to retrieve any vote at all. We don't know if this means 10 per cent of iVotes were manipulated or dropped—these verification attempts may have failed for some other reason—but we do know that the NSWEC simply didn't tell the truth at the time of the election. So iVote has proven that serious errors and problems can go unnoticed or unreported.

In 2017, during a run of iVote in Western Australia, our team found that all iVotes were being funnelled through Incapsula/Imperva, a TLS proxy service, which gave it the opportunity to read and alter votes (though it would require a

[1] `https://www.elections.nsw.gov.au/About-us/Public-interest-information/iVote-reports/Response-from-the-NSW-Electoral-Commission-to-iVot`

significant amount of work) [CEET17]. We found servers linked to this network in North America, South America, China, and Western and Eastern Europe. This service acted as a proxy for both the registration service and the voting stage, so the voter's identity could be easily linked to their vote. NSWEC and the VEC have also deployed voting or registration services using Incapsula/Imperva. In August of this year Incapsula/Imperva released a statement[2] that a breach had occurred affecting a subset of their customers for a period through September 15, 2017. No further information has been provided as to which customers it involved, nor have the Electoral Commissions confirmed or denied whether they were in the subset that were breached.

iVote's security problems are not unusual. Independent studies of similar systems have shown a pattern of similar vulnerabilities [WWIH12, SFD$^+$14]. A recent Russian e-voting system was shown to be using cryptographic primitives that took 20 minutes to break.[3]

The point here is not that iVote and its contractors are particularly bad, but that they are just like any other electronic systems: subject to bugs and security holes that can be exploited by attackers though they go unnoticed by their owners for a long time.

3 iVote 2019 and the SwissPost system

In 2019, while examining source code of the SwissPost e-voting system that had been made publicly available for testing, we found serious cryptographic errors that could affect the iVote system as well [LPT19a, LPT19b]. Although the system is different, both systems are supplied by the same vendor (Scytl) and have a lot of code in common.

The Swiss system includes a mathematical proof that the encrypted votes have been properly shuffled and honestly decrypted. We showed that this proof was not sound. There were several different ways in which Scytl, or anyone else with access to the server, could forge a "proof" that passed verification even though the votes had been manipulated.

NSWEC confirmed that the iVote system was affected by the problem but claimed that "the machine on which the mixnet runs is not physically connected to any other computer systems." This is not relevant: this is an insider attack that forges a proof of integrity even if the system is secured from the outside.

At the time, the NSWEC said: "Our processes reduce this risk as we specifically separate the duties of people on the team and control access to the machine to reduce the potential for an insider attack. Scytl is delivering a patch which will be tested and implemented shortly to address this matter." So they are saying in one paragraph that they were defending against insider attacks, while also deploying a hastily-implemented patch from a foreign provider to the core voting system within a few days of the election.

Furthermore, "not physically connected to any other computer..." does not imply that it was separated from any network. The heavily-redacted post-election report from the multinational consultancy PwC makes no mention of correcting the cryptography, but in issue 17 it says[4] "[redacted] on air-gapped

[2] https://www.imperva.com/blog/ceoblog/

[3] https://members.loria.fr/PGaudry/moscow/

(offline) computers was not disabled." We respectfully suggest that the redacted word refers to the Wireless Internet Connection, so that the supposedly air-gapped machines could actually connect to the Internet. This doesn't prove that they were connected, only that there was nothing preventing them from connecting. And if the machine on which the mixnet runs did exfiltrate its data, it could reveal how everyone voted. If it was controlled by an adversary, it could change the votes.

After having agreed that it was vulnerable to the first of the errors we found in the SwissPost shuffle proof, the NSWEC denied that iVote was affected by the second problem we identified in the same system.[5] This error concerned the decryption part of the proof—a cheating decryption service could claim to have correctly decrypted a vote but actually substitute nonsense that would not be counted, while passing verification. The problem was that not enough data was included in the cryptographic hash. Since the NSWEC had been warned already about incorrect data being included in the hash function of their decryption proof,[6] the claim that iVote was unaffected by this flaw is highly implausible.

3.1 Voter Verification

iVote contains a mechanism for allowing voters to query whether their vote was sent in to the central iVote server correctly. Although it is called the iVote 'Verification App,' it is not really a verification mechanism, because it provides no evidence that the reported vote matches the vote that will be counted. Nevertheless it might catch some kinds of accidental (or even deliberate) errors.

In 2015, the query mechanism was implemented by phone, but in 2019 voters were encouraged to use a smartphone app instead. The smartphone app was implemented by Scytl, the same company that provided the voting system.

Voters cast a vote using their web browser. At the end of the voting session, the browser was supposed to send an encrypted version of the vote the voter entered, then print a QR code[7] on the screen. If the voter didn't trust software in her web browser to cast his vote correctly, she could download the app onto her smartphone, hold its camera up to the QR code, and ask the app what vote the browser code had sent.

There are two serious problems here. First, it didn't work—the Google App Store alone contains hundreds of reports from voters who couldn't get the app to read the QR code at all. Consider that this is the only publicly-available method for assessing the fraud or error rate in iVote, and that the 2015 run suffered from a 10% failure rate. A verification failure might be the result of an innocent software bug, or it might indicate systemic electoral manipulation. There is no way to tell, but at least we can try to assess the magnitude of the problem. What was iVote's verification failure rate in 2019? There is probably no way to know.

[4] https://www.elections.nsw.gov.au/getmedia/b2280c43-a129-47ca-bd75-f9c98887736b/2019-State-Elections-iVote-review-(post-election-report)-June-17-2019-redactions-v2-3-draft-Copy_Redacted(1)

[5] https://elections.nsw.gov.au/About-us/Media-centre/News-media-releases/NSW-Electoral-Commission-iVote-and-Swiss-Post

[6] See Shürmann and Hook, p.6 https://www.elections.nsw.gov.au/getmedia/341fa362-e859-4f36-89d5-1b7385e195b7/Code-review-report_NSWEC-7-Redacted

[7] A QR code is a glorified 2-dimensional barcode. In this case, it was supposed to convey information that would help retrieve the person's vote from the server.

The second problem is even more fundamental: *this verification mechanism adds no evidence whatsoever.* If the software provider is honest and trustworthy, then the software sends the right vote the first time; if it is not trustworthy, it sends the wrong vote and then its verification app lies about what vote was sent. In neither case does the voter get any information by asking a closed-source app from the same company. Nor does she have any way to prove if it did misbehave. Even innocent programming or configuration errors, such as switching the names or positions of two candidates, could be repeated in both programs, and cause the verification step to produce what the voter expected even though the true vote was different.

We recommend not calling the app a 'verification' app. Although it might detect some accidental errors and security problems that affect only the voting machine, it does nothing to prevent calculated fraud. It would be better to tell voters honestly that they have no way to verify whether their vote was accurately recorded and included.

3.2 Source code openness

The serious cryptographic weaknesses in iVote's integrity proofs came to light only because responsible authorities in Switzerland chose to make their code available to scrutiny six months before their election. This happened to coincide with the running of the NSW election. NSWEC had not given themselves any real opportunity to learn of problems beforehand because the iVote code was made available only under a restrictive Non Disclosure Agreement that prohibited researchers sharing their findings with the public for five years. Terms like these are unacceptable to security researchers who recognise their ethical obligation to alert the public if serious errors remain that might compromise an election result—imagine knowing that a suspicious election result might have been caused by exploitation of a known vulnerability, but being contractually disallowed from notifying the affected candidate or their supporters.

Source code for iVote has now been made available, more than four months after the election, so it is now possible to inform the electoral commission of the serious problems in the code it ran many months ago. There is, obviously, no hope of fixing such errors in time for the election, which has now passed. A losing candidate would have lost the opportunity to challenge.

Some specific findings relating to the code that is now available will be shared with the committee when the 45-day non-disclosure period has passed.

The punitive clauses in the NSW electoral Act, which criminalise sharing system details and source code without permission from NSWEC, stand in stark contrast to Swiss laws mandating openness of source code and documentation (for systems that may be used by up to 100% of voters) well in advance of the election. We believe it is a direct consequence of NSW anti-openness laws that, for the third time in a row, the Australian authorities learned of serious vulnerabilities in iVote *only when the election was already running.*

Regardless of whether iVote is allowed to continue, the NSW parliament should rewrite electoral law to mandate openness of election source code and documentation, rather than punishing people who try to tell other citizens about the details of how their elections work. This would have the significant practical advantage of allowing security analysis to be performed before the election, as it is in Switzerland, rather than during the election, when it is too late to patch.

4 Discussion

iVote makes large-scale fraud possible for anyone who controls the system, and for many others with legitimate or illegitimate access to parts of the system. That fraud could be completely undetectable. Even if we didn't notice any problems, we have no idea whether the election outcome is accurate.

It is not helping voters with disabilities, nor voters who live a long way from a polling place, to pretend to be offering them a chance to vote when in fact the process is so easily manipulated that there is no reason to think their votes are accurately recorded or included at all.

There are numerous alternatives to paperless Internet voting. We could extend the early voting period to give people more time to get to the polls. We could run computers in a polling place with a voter-verifiable paper record and a risk-limiting audit. We could make candidate information available online and get voters to post or deliver a printout of their vote.

There are many reasonable options, all of them far more secure than iVote.

References

[CEET17] Chris Culnane, Mark Eldridge, Aleksander Essex, and Vanessa Teague. Trust implications of ddos protection in online elections. In *International Joint Conference on Electronic Voting*, pages 127–145. Springer, 2017.

[HT15] J Alex Halderman and Vanessa Teague. The new south wales ivote system: Security failures and verification flaws in a live online election. In *International conference on e-voting and identity*, pages 35–53. Springer, 2015.

[LPT19a] Sarah Jamie Lewis, Olivier Pereira, and Vanessa Teague. Ceci n'est pas une preuve: The use of trapdoor commitments in bayer-groth proofs and the implications for the verifiabilty of the scytl-swisspost internet voting system, 2019. https://people.eng.unimelb.edu.au/vjteague/UniversalVerifiabilitySwissPost.pdf.

[LPT19b] Sarah Jamie Lewis, Olivier Pereira, and Vanessa Teague. How not to prove your election outcome: The use of non-adaptive zero knowledge proofs in the scytl-swisspost internet voting system, and its implications for decryption proof soundness, 2019. https://people.eng.unimelb.edu.au/vjteague/HowNotToProveElectionOutcome.pdf.

[SFD$^+$14] Drew Springall, Travis Finkenauer, Zakir Durumeric, Jason Kitcat, Harri Hursti, Margaret MacAlpine, and J Alex Halderman. Security analysis of the estonian internet voting system. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pages 703–715. ACM, 2014.

[WWIH12] Scott Wolchok, Eric Wustrow, Dawn Isabel, and J Alex Halderman. Attacking the washington, dc internet voting system. In *International Conference on Financial Cryptography and Data Security*, pages 114–128. Springer, 2012.