Review of the Overseas E-voting (OSEV) system used in the Australian Capital Territory

Thomas Haines

The Australian National University, Canberra, Australia
thomas.haines@anu.edu.au

Abstract. The Australian Capital Territory (ACT) contains the Australian national capital Canberra; the territory has a 25-member legislative assembly combing both state and local government functions. The members of the assembly are elected using two electronic voting systems. The first, the EVACS system, uses Direct-Recording Electronic voting machines (DREs) to record the vast majority of ballots in physical polling-places. Overseas voters can use the Overseas E-voting system (OSEV) to vote online. In this paper we report on our review of the OSEV system and we also reflect on the transparency of the process by which the system was introduced.

1 Introduction

The Australian Capital Territory (ACT) continues to be one of the most prominent users of electronic voting in Australia. The territory has used the—Direct-Recording Electronic voting machine (DRE) based—Electronic Voting And Counting System (EVACS) since 2001; DRE based systems are not otherwise used in Australia. Building on earlier work on formally analysing voting systems [1,3,13], there has been a string of papers analysing the counting side of the EVACS system. Goré and Lebedeva [12] showed issues in real ACT elections because of errors in the EVACS counting software. This was followed by a paper from Moses et al. [17] called "No more Excuses: Automated Synthesis of Practical and Verifiable Vote-Counting Programs for Complex Voting Schemes" which showed that it was possible to produce verified and verifiable software which could be used to count the ballots in EVACS. In 2018, T Wilson-Brown highlighted important privacy issues in the EVACS system,[1] and in 2020 Conway and Teague demonstrated yet more errors in the counting software.[2] Alas, the ACT Electoral Commission, hereafter referred as the commission, has been reticent to address the issues raised by the academic community.

Despite the issues with the EVACS system, the publication of the source code at least allowed interested parties some ability to scrutinise the system; alas no longer, in the lead up to the 2020 election a new version of the EVACS

[1] https://www.abc.net.au/news/2018-08-14/voters-in-act-election-could-have-ballot-choices-identified/10115670
[2] https://github.com/AndrewConway/ConcreteSTV

system was proposed for use alongside a new online voting system called the Overseas E-voting (OSEV) system. The systems were not publicly available but the commission would, at its discretion, make them available upon the reviewer signing a Non-Disclosure Agreement (NDA). We will discuss the NDA in more detail section 3.3.

At this point, the commission was still insisting publicly (and on its website) that the source code was publicly available. Requests by academics to review the code were answered by saying the code wasn't ready for the review even though voting would be starting the following week. Freedom Of Information (FOI) requests to seek the code and audit specs were delayed because commission said that running an election was an "exceptional circumstance" for them. Finally on the 13th of November 2020, about a month after the election ended (17th October 2020), heavily redacted audit specs were released but access to code still required signing the extremely problematic NDA. In our review that follows we highlight that even the heavily redacted documents that were released were not accurate.

We formally requested access to the source code and offered to sign the NDA in early October 2020. We were notified in mid-February 2021 that the commission had declined our request "because it (was) not satisfied that the risk that the source code may be improperly accessed by others can be appropriately managed." After further discussions, we were finally able to access the source code of the OSEV system in June of 2021.

In the next subsection we will discuss what is publicly known about the OSEV system then the remainder of the paper follows in three sections. First, in Section 2 we detail the scope of the review and the findings. In Section 3, we reflect on the process surrounding e-voting in the ACT and the interplay between that process and the security of the e-voting systems and compare to the recommendations in the literature. Finally, we conclude in Section 4.

1.1 Overview of OSEV system

Public details of the OSEV system are sparse which makes it hard to provide much information without engaging in speculation or violating the NDA we signed. In the remainder of this subsection will summarise the publicly known information about the system and its security requirements.

Sources of information Publicly knowledge of the security goals and system design of OSEV is based upon the following (heavily redacted) documents as released in response to T Wilson-Brown's FOI request:[3]

OSEV Architecture Diagram v1.1
which provides a summary of the various components mentioned later in the report and their interaction [4].

[3] The FOI request can be found at `https://www.righttoknow.org.au/request/vote_secrecy_in_2020_election`

OSEV Authentication Sequence Diagram v0.6.2, OSEV Register Sequence Diagram v0.6.1, OSEV Check Sequence Diagram v0.6.0, OSEV Export Sequence Diagram v0.6.1
which provide summaries of the principal interactions between the various components [5,9,6,8].

OSEV System Design v3.0
which provides an overview of the components and the design goals [11].

OSEV Security Summary v1.4, OSEV Detailed Requirements v1.2
which detail the security requirements the system is designed to achieve [10,7].

In addition our review is based on the source code from version 4cba9731_v1_0_0_prod of the overseas e-voting system; this was made available to us upon signing a non-disclosure agreement. This source code did *not* include any documentation which provided additional insight into the system design or security goals.

Summary of security goals The (unredacted) security goals of OSEV are described vaguely and we summarise them below as they are found in the documents. We note the descriptions in the documents tends to focus on the security mechanism not the security goal.

OSEV System Design v3.0 [11] contains about seven pages of information which not been redacted. Only one page of this document covers design features such as:
- Separation of system components to distribute trust
- Not allowing voter preferences to be linked to a voter
- Vote integrity - the vote should not be tampered with
- Process integrity - the process flow of the election should be followed; for example, ballots should only be accepted if they come from registered voters and during the period that voting is open
- No direct link to existing systems such as the EVACS counting module or the system which stores the list of eligable voters

OSEV Security Summary v1.4 [10] contains about four page of information which has not been redacted. The security features emphasised, in addition to those, described above are:
- No database or storage for the Web and Verify applications to protect vote privacy
- Votes are encrypted in transit and rest
- Decryption keys are not used or stored in the online system to preserve vote privacy and prevent vote tampering

OSEV Detailed Requirements v1.2 [7] contains two pages of unredacted information about half of which pertains to security:
- Information on the use of encryption and signatures
- Only valid votes will be counted
- Requirements on the use of TLS

The main issue with description of the security goals is that no threat model is described. For example, the requirement that only valid votes be counted seems to have been “satisfied” by having the Vote storage application check that the ballots it received match the records kept by the Verify and Check applications. This only works if the vote storage application is trusted for this requirement. Similarly, saying that an application had no database works well for an honest but curious adversary but since the web application in particular is connected to the internet it seems unlikely this would prevent an active adversary from retrieving information from this component.

Summary of system components The system consists of four online components called Check, Verify, Vote storage, and Web application. The system also has a Desktop application and client side code which it serves to the voters' browsers, which we have denoted Web client in the Fig. 1.

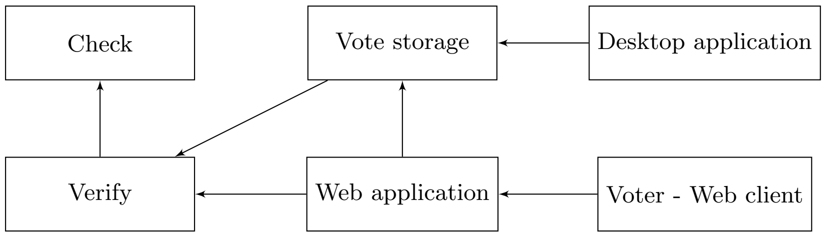

Fig. 1. Overview of components

We have provided in Fig. 1 an overview of the components with arrows denoting dependencies. For example, we denote that Web application as dependent on Verify and Vote Storage because uses the Verify API to check voter eligibility and the Vote Storage API to store votes. Fig.1 can be thought of as a simplified version of the OSEV Architecture diagram [4].

OSEV Web application: The OSEV Web application is the user-facing component of the system. This component mediates the users' interactions with the other components during registration, authentication, and voting as noted in the relevant diagrams [9,5].[4] It is also responsible for providing the web client code to the voter. To mitigate the risk to privacy and integrity of this component being compromised it does not have any storage, beyond its volatile memory; all voter identity and ballot information is stored only in the local variables of function handling the voter's request. It receives ballots from the voter over a TLS connection which it then encrypts using the public key of the OSEV system to ensure that ballots are encrypted in transit and at rest.

[4] The Authentication Sequence Diagram appears to reference a Voting Sequence Diagram which was not released in response to the FOI.

OSEV Web client: The voter uses a browser to register and vote through a website provided by the system. The web site does not directly encrypt the vote but relies upon the TLS protocol to secure the vote in transit to the OSEV Web application were it will be encrypted for storage. The web client is denoted as the voter in the relevant sequence diagrams [9,5].

OSEV Vote storage: The OSEV Vote storage is responsible for storing the ballots collected by the Web application and making them available to the OSEV Desktop application. It performs checks with OSEV Verify to ensure that the Web application does not add ballots from ineligible voters, as noted in the Export Sequence Diagram [8]. The Export Sequence Diagram shows the Vote storage system receiving the RSA private key (which can decrypt the ballots) during export process; this is a direct contradiction with the security requirement that online system not have access to the secret key. Fortunately, when looking at the code we found this was not the case and the private key is kept only on the (offline) Desktop application.

OSEV Verify and OSEV Check: OSEV Verify and OSEV Check are two components which work together to ensure that voters and ballots are authorised [5,9]; they also perform the other checks required to ensure the online part of the election runs in an orderly manner. In essence OSEV Verify serves as a stateless proxy to OSEV Check with the aim of increasing privacy. Together they: register voters for the OSEV system and check their eligibility, provide the ballot "paper" for a given voter, and key track of who has voted.

To provide this service OSEV Check interacts with other election management software outside the OSEV system, this other software is called Tiger in the OSEV Check Sequence Diagram [6]; we were not given details of this other software which hindered making conclusions about these two components.

OSEV Desktop application: The OSEV Desktop application downloads the votes at the end of the election period. These votes are then decrypted using the secret key of the OSEV system before the resulting decryptions are encrypted using the public key of the EVACS system. The encryptions under the EVACS key are added to the other votes which will then be tallied. The OSEV Desktop application, when exporting to EVACS, does not appear to include any information that would allow modifications of the ballots to be detected.

2 Review of the OSEV system

Our review was based on version 4cba9731_v1_0_0_prod of the Overseas E-voting system source code. It also draws upon the documents detailed previously released through a Freedom of Information (FOI) request by T Wilson-Brown. Overall, we find that the code in its current state offers no integrity advantage over a single web application, and little privacy advantage; in other words, the security goal of distributing trust by separating the system components was not achieved. The security of the system relies upon procedural mechanisms which were not open to scrutiny in the course of this review.

In our review it very quickly became apparent that we were not going to be able to assess the deployed security of the system in detail due to lack of information; specifically, we lacked information about the interaction of the OSEV system with wider election management software and more crucially about the procedural mechanism around it's deployment. We therefore focused our review on assessing the distribution of trust among the system components and the use of cryptography. We did not, and could not, review the procedural mechanisms. Nor did we test exhaustively for the presence of buffer overflows, input sanitation errors, etc. All the findings detailed in this section and the next draw at least partially on the materials subject to the NDA.

2.1 Methodology and Scope

This review is based on 4cba9731_v1_0_0_prod of the Overseas E-voting system source code. The source code was delivered in a compressed file of around 80MB. We did not receive any documentation with the exception of some (outdated) README files.

We scoped our analysis to the code which handled ballots either by encrypting, decrypting, storing, or determining validity. Outside the scope were, in general, all other code. For example, we reviewed the parts of OSEV Verify and OSEV Check used by the Vote storage application when checking which ballots are valid but did not analysis the parts of OSEV Check and OSEV Verify which ensured ballots came from eligible voters. Also outside of scope was the external libraries used by the system.

Our review methodology relied upon manual code review using an IDE, which supported the relevant language. We were unable to build or execute the system as whole; however, we did isolate sections of the code for which we constructed test suites using a common unit testing framework. We assessed the code with respect to integrity and privacy goals of the system, which we summarised in 1.1.

We were unable to assess all procedural mechanisms not contained within the code. For example, the mechanism by which it was ensured that the Web application and OSEV Verify had no access to storage was not in scope.

2.2 Areas of Concern

1. The code and architecture documents show a clear intention to distribute trust over various components. Avoiding a single point of failure is a very desirable property for an e-voting system – some might say a necessary one – but the current system falls short of achieving this on a few points.
   (a) There are single points of failure for both privacy and integrity if a key component is compromised before or during the election, particularly the Web application as we will discuss in point 4.

   (b) Several components accept the input of other components without adequate validation, as detailed in the following concerns.

   Building a secure distributed system is a challenging task and the electoral commission will need to draw on additional expertise if they wish to achieve this.
2. The code was not in a polished state, which hinders analysis both by external parties and the internal development team. Specifically, the code contained unused legacy material which was hard to distinguish from what was actually relevant. We suggest that comments accompanying the code, including the README document, be kept clean and updated.
3. The version of the source code we were given access to did not meet the requirements listed in the documents released as a result of T Wilson-Brown's FOI request. In one specific instance, we were informed that the requirement had been removed. Assuming that the code we were given reflects the code used in the election on the 17th of October 2020, we are disturbed that documents released in response to the FOI request (on the 13th of November 2020) included requirements which had been removed.
4. The Web application learns the user's identity at registration and the vote when the user votes. It is, therefore, a single point of failure for privacy. The Web application can drop or modify ballots without detection; however, the checks performed by vote storage prevent the Web application from stuffing the ballot box. *The commission has assured us that these attacks are mitigated by procedural mechanisms which are outside the scope of this review and unassessable based on the material made available to us.*
5. The OSEV Web client depends on TLS to safeguard the privacy of the vote. We are concerned that the procedural mechanisms used by the commission, for example to protect against denial-of-service attacks, may allow a third party to read votes in transit. A similar issue occurred in iVote system [2]. *The commission has assured us that there is no attack here but we are unable to verify this without knowledge of the procedural mechanisms in place.*
6. The Vote storage application is a single point of failure for integrity since the Desktop application does not check the consistency of the Vote storage's output with other components.

   The commission initially claimed that no attack on integrity was possible because the Voter storage system did not know the (public) key used to encrypt the votes. We communicated to the commission that knowledge of the (public) key was not required to modify the votes for the encryption scheme they were using. They have now acknowledged the issue and "will work to address it in the future deployments of OSEV." *The commission has assured us that the attack was nevertheless mitigated by procedural mechanisms which are outside the scope of this review.*
7. The Verify and Check applications have been separated from each other, rather than existing as a single component, with the intention of improving privacy. In our judgement, the separation of OSEV Verify from OSEV Check does not appear to meaningfully improve the security of the system. This is certainly the case at present since the Web application is a single point

of failure for privacy already; we suspect this would remain the case even if the most obvious issues are fixed. However, due to the interaction of these components with other systems—which we were not given access to—we are not certain. Further investigation would be required to judge precisely what security is provided by the current separation; such an investigation would be complicated by the lack of a clear and detailed security goal.

2.3 Reflections and Recommendations:

Based on the areas of concern above we made several recommendations to the commission.

Due on our concern about the very under-defined security model, we strongly recommended the commission carefully review the security requirements to ensure that they are satisfied they are sufficient. Given that the commission may lack the capability to adequately do this in-house we encouraged them to seek external advice. Based on the issues currently present in the system, we strongly recommended the commission seek support from members of the public with relevant expertise to ensure they are aware of, and can address, issues with the system.

Following up on our first recommendation and desiring public transparency about the level of security, we encouraged the commission to release an unredacted document which clearly articulates the high level security properties they wish to achieve.

In our final recommendation we were again concerned about public transparency about security, we encouraged the commission to make sufficient information and parts of the system available to public scrutiny, to allow interested members of the public to check that the high level security properties are achieved.

2.4 Review results

The results of our review were released under section 5 of the non-disclosure agreement which allows publication of findings after a certain period. All findings in this report were disclosed to the commission no later than the 10 August 2021.

We have deliberately kept our results section short and avoided specifics as much as possible since our Non-Disclosure Agreement (NDA) says we may not include any part of the source code. We will discuss the NDA in Sec. 3.3 and specifically how we choose what information to try and make public.

Detailed recommendations arising from the review provided to the commission

The OSEV Desktop application should validate the received ballots to the greatest extent possible. Specifically, it should check that the data (encrypted votes) provided by OSEV Vote storage is consistent with OSEV Web application, Verify and Check.

This is will detect attempts by OSEV Vote storage component to tamper with the votes it receives.

Integrity The system should be constructed so that no ballot can be added, modified, or dropped without detection provided at least one component, of the five, is honest (ideally this should also protect against malware on the voter's computer).

Privacy The system should be constructed so that no information (beyond seeing a randomly ordered list of all ballots) is leaked provided the encrypting component and at least one other component is honest. Ideally, encryption should occur on the voter's device; barring this the encrypting component should not have any information about the identity of the voter (other than a token and what could be discerned from the user's connection).

Our second and third recommendations seems to capture the distribution of trust over the components which the high level descriptions of OSEV provided by the commission seem to envision but the system does not achieve. Realising this level of security would require a significant redesign; for example, at present the issues with the Web Application component seem to be unsolvable without introducing a cast-as-intended mechanism to the system.

Detailed comments on the documentation after reviewing the source code

- The documentation, and specifically requirement OSEV64 (from OSEV Detailed Requirements), requires that votes are signed by the Web application, but we could find no evidence of this occurring.

  In response to our report on this matter, the commission notified us that this requirement was actually removed. The given justification for removing the requirement was that the asymmetric encryption, of the votes, made this unnecessary. We are unsure when the requirement was removed but given that the asymmetric encryption does not prevent vote tampering we strongly recommend the commission reintroduce this requirement.
- Comment 3.a in the OSEV System Design document which appears under the section "Vote Integrity" says "Vote preferences cannot be read by an unauthorised party because they are encrypted using an RSA asymmetric algorithm so that they can only be decrypted by a key not stored by the system and instead held by Elections ACT." relates to privacy rather than integrity. The general confusion about basic security properties evidenced by the vendor and commission is a central reason why we conjecture the lack of transparency is likely to hide further vulnerabilities.

  In general it seems that several components could add, edit, or drop votes without detection; as we detailed in Sec. 2.2. We do not preclude that there are other mechanisms, which might catch the tampering, of which we are unaware. However, in the scope of the documents released, the integrity of the system is at best poor unless all components are behaving properly.

3 Reflections on the process

Two recent papers [14,16] by Haenni et al. and, Haines and Rønne comment on best practice for processes around e-voting systems. Below, we list selected best practices and contrast to the processes around the OSEV system. We have done this because we conjecture that specific vulnerabilities, like those we listed in the previous section, are symptoms of poor processes; we further conjecture that focusing on improving the processes is the best way to deliver secure systems. We cannot test this conjecture in this paper alone but our hope is that if papers like ours report not only vulnerabilities but also issues with processes then in several years we should have sufficient data to assess the claim.

3.1 Selected principles from CHVote: Sixteen Best Practices and Lessons Learned

Modelling the Electoral Systems The first principle is the importance of properly modelling the electoral system; this is required to provide a proper level of abstraction of the electoral process when designing the voting protocol.

The fact that ACT elections are for a single race, with the occasional exception of a referendum, makes creating a model of the election system relatively straightforward; our review indicates that the OSEV system complies with this principle.

Modelling the Electorate The second principle is strongly related to the first; the model of electoral system needs to properly and succinctly capture which voters are eligible to vote in which races.

Similar to the modelling the election system, modelling the electorate is straight forward in the ACT since all voters are eligible to vote on all issues, of which there is normally only one; our review indicates that the OSEV system complies with this principle.

Cryptographic Building Blocks A correct choice of cryptographic building blocks is essential to distribute the trust over multiple components. This principal highlights not only the need to select the appropriate building blocks from the literature but also the the importance of clearly documenting which have been chosen and why.

It does not seem that the vendor and the commission had a clear idea of the cryptographic building blocks available to them, or the functionality of those building blocks. This has resulted in a system where it is unclear what security is achieved; our review indicates that the OSEV system does not comply with this principle.

Cryptographic Parameters This principle highlights the importance of correctly and consistently chosen security parameters, as well documenting these well.

The OSEV system used existing libraries to implement the cryptographic building blocks. The choice of parameters was largely handled by these libraries. This seemed to work reasonable well, with the exception that the cryptographic building blocks did not provide the functionality that the vendor and commission believed it did; our review indicates that the OSEV system does comply with this principle but highlights that compliance here without good choices of cryptographic building blocks does not provide the required security.

Parties and Communication This principle highlights the importance of clearly defining the responsibilities, abilities, goals, and trust assumptions for each participant in the protocol.

The OSEV system defines reasonably well the protocol participants and communication. However, the functionality required of each participant in the context of an overall security model, or lack thereof, was missing; our review indicates that the OSEV system does not comply with this principle.

Protocol Structure and Communication Diagrams This principle highlights the importance of precise and comprehensive description of the voting protocol.

The commission and the vendor produced a number of protocol and communication diagrams but the versions released to the public were heavily redacted; our review indicates that the OSEV system does not comply with this principle.

Pseudo-Code Algorithms This principle suggests presenting pseudo-code algorithms for every computational task in the protocol. This maximises the technical depth of the specification.

Pseudo-code algorithms were not available in the information made publicly available. This would have been very useful in analysing the protocol; our review indicates that the OSEV system does not comply with this principle.

Implementation of Pseudo-Code Algorithms This principle encourages implementing the system so that the alignment between the implementation and pseudo-code algorithms in specification are clear.

The lack of Pseudo-Code algorithms for OSEV put extra pressure on the code to be clear as to its purpose. In many cases, it was unclear what code was doing and how key requirements were met. This is was not helped by certain key requirements being achieved by intervention from outside the system; our review indicates that the OSEV system does not comply with this principle.

Cryptographically Relevant Code This principle encourages separating the cryptographically relevant and cryptographically irrelevant components, instead linking them over suitable interfaces.

The reliance on existing libraries to implement the cryptographic building blocks in OSEV turned out to be problematic. It seems clear from the documents that neither the vendor or the commission had a clear view of what cryptographic building blocks were being used except at a very high level. Crucially, the sufficiency of security properties of these building blocks in the context of the protocol were not considered; our review indicates that the OSEV system does not comply with this principle.

Transparency This principle highlights that transparency around the protocol is fundamental to the success of an e-voting project.

The lack of transparency around the OSEV system and the process is of great concern. The lack of publicly available high level security goals and sufficient information to verify that those goals are met means that stakeholders in the election have no means to assess the suitability of the system; our review indicates that the OSEV system does not comply with this principle.

Verifier The lack of any clear notation of a verifier, or verifiers, is a major problem in the OSEV system. As highlighted in our Areas of Concern (Sec. 2) the lack of validation the components perform on the input they receive from other components is one of the major reasons the system is not secure if one or more components are compromised; our review indicates that the OSEV system does not comply with this principle.

3.2 Principles from New Standards for E-voting Systems

Haines and Rønne [16] give nine high level principles about e-voting systems which focus heavily on the systems themselves and the process directly around them. *None* of their principles were met by the OSEV system and we give details below:

Clear claims The documentation accompanying the system should be clear about what security properties the system—and its sub-components—claim to achieve.

There are no clear security claims about the OSEV system due mainly to the lack of a threat model.

Thorough documentation The documentation—and source code comments—should be comprehensive, clear, correct, and consistent.

The OSEV documentation, while extensive, is heavily redacted and focuses on functionality not security.

Minimality The source code provided should be minimal; it should contain only code related to the system under review.

The code base includes out-of-date material which hindered analysis.

Buildable The released source code should be easy to build. Preferably it should come with a configuration using a standard tool, such as Maven. The system should not depend on proprietary libraries which have not been released.

The instructions on building the code included with the code did not work.

Executable The system, once built, should be executable. The intended execution flow of the code should be clear either from the documentation or tests.

Since the code was not buildable it was not executable.

Exportable It should be possible to export test vectors into a well defined format for testing with an independent verifier.

Since the system lacked a notation of a verifier it was not possible to export the data required for the, non-existent, verifier.

Consistent documentation and source The source code and the documentation should correspond to each other.

The alignment between the heavily redacted documentation and source was not clear; in several cases, there were clear gaps. For example, see the discussion in Sec. 2.4.

Regularly updated The open source variant of the system should be regularly updated so that experts can check that previous bugs are correctly fixed.

The lack of transparency around the code base makes it difficult to assess how regularly the code was updated. There were several cases were parts of the system were out-dated, which we discovered in discussion with the commission.

Minimal restriction on disclosure The restrictions on the disclosure of vulnerabilities should be minimal.

The NDA required to access the code is unclear as to what findings can be published and when, with seemingly punitive conditions for breaking the vaguely worded agreement.

3.3 Transparency and the NDA

The non-disclosure agreement required the reviewer to accept legal liability for all claims, costs and expenses made against the territory, its employees and agents as a result of the the reviewer breaching the NDA. The NDA did not make clear what information reviewers would eventually be allowed to make public or when they would be allowed to make it public; the waiting period was 60 days but it was unclear from when. The NDA did make clear that not even part of the source code could be made public. Therefore, we have withheld all information which would allow parts of the code to be reconstructed. We have furthermore avoided mentioning what language the system is implemented and what libraries it depends on.

During the course of our investigation it became clear that we were not going to be able to precisely analyse the claims around system based on the (lack of) information we had been given; it was also clear that the implementation of the system could not provide the distributed trust that the design document called for. Based on this we decide to limit our report primarily to high level issues and avoid publishing more specific, and speculative, points.

In making our initial public disclosure [15] we made not attempt to interpret what material we were allowed to disclose. Rather, we provided the commission with a draft report. We requested that either they explicitly give us permission us to publish that report or make public why our report should be withheld.

To our knowledge we are the only party to sign the NDA and get access to the source code. We were only willing to do this because of the academic freedom policy of our university which enshrines the right to discuss, and research and to disseminate and publish the results of our research. This allowed us to conduct the research as part of our employment and not as an individual. In discussion with our colleagues, at other academic institutions but particularly in industry, we are aware of several highly qualified individuals who wished to provide feedback on the system but were unwilling to take the risk; their understandable decision means the NDA has cost the voters in the ACT invaluable feedback on the system used for their elections.

4 Conclusion

The Overseas E-voting (OSEV) system used in the Australian Capital Territory is an excellent case study in how not to do online voting. The system does not follow identified best practice (Sec. 3) and its security and security goals are not open to public scrutiny (Sec. 2). Furthermore, it seems clear that neither the vendor nor the commission have a clear understanding of the security level the system achieves.

References

1. Beckert, B., Goré, R., Schürmann, C.: On the specification and verification of voting schemes. In: VoteID. Lecture Notes in Computer Science, vol. 7985, pp. 25–40. Springer (2013)
2. Culnane, C., Eldridge, M., Essex, A., Teague, V.: Trust implications of ddos protection in online elections. In: E-VOTE-ID. Lecture Notes in Computer Science, vol. 10615, pp. 127–145. Springer (2017)
3. Dawson, J.E., Goré, R., Meumann, T.: Machine-checked reasoning about complex voting schemes using higher-order logic. In: VoteID. Lecture Notes in Computer Science, vol. 9269, pp. 142–158. Springer (2015)
4. Digital Elections Pty Ltd and Blitzm Systems: OSEV architecture diagram v1.1. `http://www.elections.act.gov.au/__data/assets/pdf_file/0004/1659811/OSEV-Architecture-Diagram_v1_1.pdf` (2020), accessed: 2022-07-14
5. Digital Elections Pty Ltd and Blitzm Systems: OSEV authentication sequence diagram v0.6.2. `http://www.elections.act.gov.au/__data/assets/pdf_file/0005/1659812/OSEV-Authentication-Sequence-Diagram.pdf` (2020), accessed: 2022-07-14
6. Digital Elections Pty Ltd and Blitzm Systems: OSEV check sequence diagram v0.6.0. `http://www.elections.act.gov.au/__data/assets/pdf_file/0006/1659813/OSEV-Check-Sequence-Diagram.pdf` (2020), accessed: 2022-07-14
7. Digital Elections Pty Ltd and Blitzm Systems: OSEV detailed requirements v1.2. `http://www.elections.act.gov.au/__data/assets/pdf_file/0009/1659816/OSEV-Detailed-Requirements-v1_2.pdf` (2020), accessed: 2022-07-14
8. Digital Elections Pty Ltd and Blitzm Systems: OSEV export sequence diagram v0.6.1. `http://www.elections.act.gov.au/__data/assets/pdf_file/0010/1659817/OSEV-Export-Sequence-Diagram.pdf` (2020), accessed: 2022-07-14
9. Digital Elections Pty Ltd and Blitzm Systems: OSEV register sequence diagram v0.6.1. `http://www.elections.act.gov.au/__data/assets/pdf_file/0012/1659819/OSEV-Register-Sequence-Diagram.pdf` (2020), accessed: 2022-07-14
10. Digital Elections Pty Ltd and Blitzm Systems: OSEV security summary v1.4. `http://www.elections.act.gov.au/__data/assets/pdf_file/0007/1659823/OSEV_Security_Summary_v1_4_.pdf` (2020), accessed: 2022-07-14

11. Digital Elections Pty Ltd and Blitzm Systems: OSEV system design v3.0. `http://www.elections.act.gov.au/__data/assets/pdf_file/0005/1659821/OSEV-System-Design-v3_0.pdf` (2020), accessed: 2022-07-14
12. Goré, R., Lebedeva, E.: Simulating STV hand-counting by computers considered harmful: A.C.T. In: E-VOTE-ID. Lecture Notes in Computer Science, vol. 10141, pp. 144–163. Springer (2016)
13. Goré, R., Meumann, T.: Proving the monotonicity criterion for a plurality vote-counting program as a step towards verified vote-counting. In: EVOTE. pp. 1–7. IEEE (2014)
14. Haenni, R., Dubuis, E., Koenig, R.E., Locher, P.: Chvote: Sixteen best practices and lessons learned. In: E-VOTE-ID. Lecture Notes in Computer Science, vol. 12455, pp. 95–111. Springer (2020)
15. Haines, T.: Review of the overseas e-voting (OSEV) system used in the Australian Capital Territory. Tech. rep., Australian National University (2022)
16. Haines, T., Rønne, P.B.: New standards for e-voting systems: Reflections on source code examinations. In: Financial Cryptography Workshops. Lecture Notes in Computer Science, vol. 12676, pp. 279–289. Springer (2021)
17. Moses, L.B., Goré, R., Levy, R., Pattinson, D., Tiwari, M.: No more excuses: Automated synthesis of practical and verifiable vote-counting programs for complex voting schemes. In: E-VOTE-ID. Lecture Notes in Computer Science, vol. 10615, pp. 66–83. Springer (2017)