Robis Elections, Inc. AskED ePollbook CA Electronic Poll Book System Source Code Review Test Report for California

ROB-18001SCRTR-01

Prepared for:

Vendor Name	Robis Elections, Inc.
Vendor System	AskED ePollbook CA

Prepared by:

4720 Independence St.
Wheat Ridge, CO 80033
303-422-1566
www.SLICompliance.com

Accredited by the Election Assistance Commission (EAC) for Selected Voting System Test Methods or Services

Revision History

Date	Release	Author	Revision Summary
August 28th, 2018	1.0	*J. Panek*	Initial Release
September 7th, 2018	1.1	*J. Panek*	Minor updates made

Disclaimer

The information reported herein must not be used by the client to claim product certification, approval, or endorsement by NVLAP, NIST, or any agency of the Federal Government.

Trademarks

- SLI is a registered trademark of SLI Compliance.
- All products and company names are used for identification purposes only and may be trademarks of their respective owners.

TABLE OF CONTENTS

OVERVIEW **4**
REFERENCES 4
REVIEW PROCESS **4**
SECURITY CODE REVIEW PROCESS 4
INTEGRITY CODE REVIEW PROCESS 4
VULNERABILITY CODE REVIEW PROCESS 5
REVIEW RESULTS **7**
SECURITY SOURCE CODE REVIEW ANALYSIS 7
INTEGRITY SOURCE CODE REVIEW ANALYSIS 7
VULNERABILITY SOURCE CODE REVIEW ANALYSIS 9
FINDINGS **9**
SECURITY SOURCE CODE REVIEW DISCREPANCIES 9
INTEGRITY SOURCE CODE REVIEW DISCREPANCIES 10
VULNERABILITY SOURCE CODE REVIEW DISCREPANCIES 10
CONCLUSION **10**

OVERVIEW

This Test Report details the Security, Integrity, and Vulnerability review of the source code of the **Robis AskED ePollbook CA** electronic poll book system.

References

The following key documents were used in preparing this test plan.

1. California Electronic Poll Book Regulations
2. Objective-C Style Guide: http://google.github.io/styleguide/objcguide.html
3. Visual Basic Coding Conventions: https://docs.microsoft.com/en-us/dotnet/visual-basic/programming-guide/program-structure/coding-conventions
4. 159.355: Concurrent Programming & Operating Systems: http://www-ist.massey.ac.nz/csnotes/355/pascalfc/styleguide.html
5. SQL Style Guide https://www.sqlstyle.guide/

REVIEW PROCESS

Security Code Review Process

A manual security review of the source code was conducted to analyze the Objective C, VB.Net, T-SQL, and Pascal code for findings against the following requirements:

- Evaluation of the use and correct implementation of cryptography and key management.
- Search for embedded, exploitable code (such as “Easter eggs”) that can be triggered to affect the system.
- Evaluation of the likelihood of security failures being detected.
  - Are audit mechanisms reliable and tamper resistant?
  - Is data that might be subject to tampering properly validated and authenticated?
- Evaluation of the risk that a user can escalate his or her capabilities beyond those authorized.

Integrity Code Review Process

A manual Integrity review of the source code was conducted to analyze the Objective C, VB.Net, T-SQL, and Pascal code for findings against the following requirements:

- Search for use of runtime scripts, instructions, or other control data that can affect the operation of security relevant functions or the integrity of the data.
- Analysis of the program logic and branching structure.
- Adherence to other applicable coding format conventions and standards including best practices for the coding language used, and any IEEE, NIST, ISO or NSA standards or guidelines which the contractor finds reasonably applicable.
- Evaluation of whether the design and implementation follow sound, generally accepted engineering practices. Is code defensively written against:
  - Bad data;
  - Errors in other modules;
  - Changes in environment;
  - User errors; and
  - Other adverse conditions.
- Evaluation of whether the system is designed in a way that allows meaningful analysis, including:
  - Is the architecture and code amenable to an external review (such as this one)?
  - Could code analysis tools be usefully applied?
  - Is the code complexity at a level that it obfuscates its logic?
- Analysis of error and exception handling.

Vulnerability Code Review Process

A manual Vulnerability review of the source code was conducted to analyze the Objective C, VB.Net, T-SQL, and Pascal code for findings against the following requirements:

- Search for exposures to commonly exploited vulnerabilities, such as buffer overflows, integer overflow, and inappropriate casting or arithmetic.
- Evaluation of potential vulnerabilities and related issues (code quality and standards compliance), considering that an exploitable issue in a component that is not in itself security relevant could be used to subvert more critical data. This is an issue whenever the architecture of the system does not provide strong separation of the components.
- Search for dynamic memory access features which would permit the replacement of certificated executable code or control data or insertion of exploitable code or data.

Should any vulnerability be discovered, SLI will identify the particular requirement applicable to each vulnerability.

To the extent possible, reported vulnerabilities will include an indication of whether the exploitation of the vulnerability would require access by:

- Voter: Usually has low knowledge of the Electronic Poll Book System’s software and/or hardware design and configuration. Some may have more advanced knowledge. May carry out attacks designed by others.
- Poll worker: Usually has low knowledge of the Electronic Poll Book System’s software and/or hardware design and configuration. Some may have more advanced knowledge. May carry out attacks designed by others. They have access to the software and/or hardware for up to ten days, but all physical security has been put into place before the machines are received.
- Elections official insider: Usually has wide range of knowledge of the Electronic Poll Book System’s software and/or hardware design and configuration. May have unrestricted access for long periods of time. Their designated activities include:
    - Set up and pre-election procedures;
    - Election operation;
    - Post-election procedures; and
    - Archiving and storage operations.
- Vendor insider: Usually has great knowledge of the Electronic Poll Book System’s software and/or hardware design and configuration. They have unlimited access to the Electronic Poll Book System’s software and/or hardware before it is delivered to the purchaser and, thereafter, may have unrestricted access when performing warranty and maintenance service and when providing election administration services.

SLI will not verify or demonstrate exploitability of the vulnerability but the report of the vulnerability will identify factors involved in the exploitation.

Any vulnerability theories developed by the source code review team members shall, to the extent possible, be referred to the Secretary of State staff.

SLI does not verify or demonstrate exploitability of the vulnerability.

Any vulnerability theories developed by the source code review team members shall, to the extent possible, be referred to the Secretary of State staff. The review process for the code base incorporated the best effort within the time allowed to find and report observations for the above categories. As such, it is understood that there may be undetected vulnerabilities in these categories.

REVIEW RESULTS

Security Source Code Review Analysis

SLI conducted a Security source code review of the **Robis AskED ePollbook CA** electronic poll book system's Objective C, VB.Net, T-SQL, and Pascal source code for compliance with the California Electronic Poll Book Regulations.

The source code was reviewed and evaluated for the use and correct implementation of cryptography and key management.

- The expected outcome for this review was that no issue would be found.
- The actual outcome for this review was a determination that no issue was found.

The source code was reviewed for embedded, exploitable code (such as "Easter eggs") that can be triggered to affect the system.

- The expected outcome for this review was that no issue would be found.
- The actual outcome for this review was a determination that one potential issue for the VB.Net source code for the use of T4 files was found.

The source code was evaluated for the likelihood of security failures being detected in the following two areas:

1. Are audit mechanisms reliable and tamper resistant?
2. Is data that might be subject to tampering properly validated and authenticated?

- The expected outcome for this review was that no issue would be found.
- The actual outcome for this review was a determination that that no issue was found.

The source code was evaluated for the risk that a user could escalate his or her capabilities beyond those authorized.

- The expected outcome for this review was that no issue would be found.
- The actual outcome for this review was a determination that that no issue was found.

Integrity Source Code Review Analysis

SLI conducted an Integrity source code review of the **Robis AskED ePollbook CA** electronic poll book system's Objective C, VB.Net, T-SQL, and Pascal source code for compliance with the California Electronic Poll Book Regulations.

The source code was reviewed for use of runtime scripts, instructions, or other control data that can affect the operation of security relevant functions or the integrity of the data.

- The expected outcome for this review was that no issue would be found.
- The actual outcome for this review was a determination that no issue was found.

The source code was analyzed for program logic and branching structure.

- The expected outcome for this review was that no issue would be found.
- The actual outcome for this review was a determination that no issues were found in the logic and branching.

The source code was reviewed for adherence to other applicable coding format conventions and standards including best practices for the coding language used, and any IEEE, NIST, ISO or NSA standards or guidelines which the contractor finds reasonably applicable.

- The expected outcome for this review was that no issue would be found.
- The actual outcome for this review was a determination that no issues were found.

The source code was evaluated for whether the design and implementation follow sound, generally accepted engineering practices. Is code defensively written against:

1. Bad data;
2. Errors in other modules;
3. Changes in environment;
4. User errors; and
5. Other adverse conditions.

- The expected outcome for this review was that no issue would be found.
- The actual outcome for this review was a determination that one issue was found for the Objective C code related to the self-destruct file and potential changes in environment and user error.

The source code was evaluated for whether the system is designed in a way that allows meaningful analysis, including:

1. Is the architecture and code amenable to an external review (such as this one)?
2. Could code analysis tools be usefully applied?
3. Is the code complexity at a level that it obfuscates its logic?

- The expected outcome for this review was that no issue would be found.
- The actual outcome for this review was a determination that there were no issues found.

The source code was analyzed for error and exception handling.

- The expected outcome for this review was that no issue would be found.
- The actual outcome for this review was a determination of an issue that the database does not always send an error message when there are failures.

Vulnerability Source Code Review Analysis

SLI conducted a Vulnerability source code review of the **Robis AskED ePollbook CA** electronic poll book system's Objective C, VB.Net, T-SQL, and Pascal source code for compliance to the California Electronic Poll Book Regulations.

The source code was reviewed for exposures to commonly exploited vulnerabilities, such as buffer overflows, integer overflow, and inappropriate casting or arithmetic.

- The expected outcome was that no issue would be found.
- The actual outcome was a determination that no issues were found.

The source code was reviewed for evaluation of potential vulnerabilities and related issues (code quality and standards compliance), considering that an exploitable issue in a component that is not in itself security relevant could be used to subvert more critical data. This is an issue whenever the architecture of the system does not provide strong separation of the components.

- The expected outcome for this review was that no issue would be found.
- The actual outcome for this review was a determination that no issues were found.

The source code was evaluated for dynamic memory access features which would permit the replacement of certificated executable code or control data or insertion of exploitable code or data.

- The expected outcome for this review was that no issue would be found.
- The actual outcome for this review was a determination that no issues were found.

FINDINGS

This section summarizes any findings from the **Robis AskED ePollbook CA** Security, Integrity and Vulnerability Objective C, VB.Net, T-SQL, and Pascal source code review.

Security Source Code Review Discrepancies

- No issues were found in the **Robis AskED ePollbook CA** electronic poll book system's Objective C, T-SQL, and Pascal security source code review.
- One issue was found in the **Robis AskED ePollbook CA** electronic poll book system's VB.Net source security source code review.

  - File: *Agent_source_code_2.0.31.3/AskED.Bootstrap.Win/AskED.Bootstrap.Data/AskEDWebModel.Context.tt* is a T4 file used to generate a text file on demand. This is inherent to the .NET platform and allows creation of a text file based on abstract string input. These files could potentially be used by a vendor insider to embed malicious code.

Integrity Source Code Review Discrepancies

- No issues were found in the **Robis AskED ePollbook CA** electronic poll book system's VB.Net, T-SQL, and Pascal integrity source code review.
- One issue was found in the **Robis AskED ePollbook CA** electronic poll book system's Objective C integrity source code review.
  - File: *Agent_source_code_2.0.31.3/AskED.Bootstrap.Win/AskED.Bootstrap.Housekeeping/selfDestruct.cs* is a self-destruct mechanism responsible for erasing voter data from the poll book database. There is a default time set before data is erased which can be bypassed by an election official. If the parameter is either set incorrectly by the vendor or modified by an election official, data could be lost unexpectedly or at an inopportune time.

Vulnerability Source Code Review Discrepancies

- No issues were found in the **Robis AskED ePollbook CA** electronic poll book system's Objective C, VB.Net, T-SQL, and Pascal vulnerability source code review.

CONCLUSION

For the security, integrity, and vulnerability source code reviews of the **Robis AskED ePollbook CA** electronic poll book system code base, one security issue and one integrity issue were found.