Los Angeles County
Voting Solutions for All People (VSAP) 3.0
Software Test Report
for California

CAF-21006-SCRTR-01

Vendor Name	Los Angeles County
Vendor System	VSAP 3.0

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
January 26th, 2022	1.0	*B. Roberson* *M. Santos*	Initial Release
February 18th, 2022	1.1	*B. Roberson* *M. Santos*	Updates for CASOS comments

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

INTRODUCTION

This report outlines the testing SLI Compliance (SLI) followed when performing Software Testing on the **Los Angeles County Voting Solutions for All People 3.0** (**VSAP 3.0**) voting system against the California Voting System Standards (CVSS).

Coding languages involved in the **VSAP 3.0** application are shown in Table 1.

*Table 1 –**VSAP 3.0** System Languages*

Languages		
Java	JavaScript	JSX
C/C++	Make	CSS
Go	SQL	Bash
Bourne Shell/Bourne Again Shell	Python	YAML
SASS	HTML	Assembly

Source Code Review tools utilized by SLI included:

- ExamDiff Pro: a commercial application used to compare revised code to previously reviewed code
- Understand: a commercial application to perform automated review of source code.
- CheckStyle: a commercial application to perform automated review of source code

REVIEW SPECIFICATIONS

The following are the specifications for source code testing conducted on the **VSAP 3.0**.

Software Test Review

The **VSAP 3.0** includes proprietary software, the code base was tested to the applicable CVSS requirements.

Review of the code included:

- Evaluating adherence to the applicable standards in sections 5 and 7 of the CVSS.
- Evaluating adherence to other applicable coding format conventions and standards including best practices for the coding language used.
- Analyzing the program logic and branching structure.

- Evaluating whether the system is designed in a way that allows meaningful analysis, including:
  - Whether the architecture and code are amenable to an external review
  - Whether code analysis tools can be usefully applied
  - Whether the code complexity is at a level that obfuscates its logic

Security considerations reviewed against the code base included:

- Searching for exposures to commonly exploited vulnerabilities.
- Evaluating the use and correct implementation of cryptography and key management.
- Analyzing error and exception handling.
- Evaluating the likelihood of security failures being detected including:
  - Whether audit mechanisms are reliable and tamper resistant
  - Whether data that might be subject to tampering is properly validated and authenticated
- Evaluating the risk that a user can escalate his or her capabilities beyond those authorized.
- Evaluating the design and implementation to ensure that sound, generally accepted engineering practices are followed, checking to verify that code is defensively written against:
  - Bad data
  - Errors in other modules
  - Changes in environment
  - User errors
  - Other adverse conditions
- Evaluating for embedded, exploitable code (such as “Easter eggs”) that can be triggered to affect the system.
- Evaluating the code for dynamic memory access features which would permit the replacement of certificated executable code or control data or insertion of exploitable code or data.
- Evaluating the code 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.

REVIEW RESULTS

Discrepancies

Discrepancies are reported such that the California Secretary of State has a basis for evaluating the extent to which the source code meets applicable standards.

VSAP 3.0 Software Test Review

Software considerations reviewed against the source code included:

- Evaluate adherence to the applicable standards in sections 5 and 7 of the CVSS
  - The expected outcome for this review was that no issue would be found.
  - The actual outcome for this review was a determination that issues were found in the following areas:
    - Known Language Vulnerability, one instance (CVSS 5.2.8.b.v)
    - Incomplete or Missing Header Comments, multiple instances (CVSS 5.2.6.a-h)
    - Dead Code, one instance (CVSS 5.2.7.e)
- Evaluate adherence to other applicable coding format conventions and standards including best practices for the coding language used
  - The expected outcome for this review was that no issue would be found.
  - The actual outcome for this review was a determination that multiple instances of Incomplete or Missing Header Comments (CVSS 5.2.8.b.v) were insufficient.
- Analyze the 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.
- Evaluate whether the system is designed in a way that allows meaningful analysis, including:
  - Whether the architecture and code are amenable to an external review
  - Whether code analysis tools can be usefully applied
  - Whether the code complexity is at a level that 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 no issue was found.

Security considerations reviewed against the code base included:

- Evaluate 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.
- Analyze 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 that no issue was found.
- Evaluate the likelihood of security failures being detected including:
  - The expected outcome for this review was that audit mechanisms would be determined to be reliable and tamper resistant, and that any data that might be subject to tampering is properly validated and authenticated.
  - The actual outcome for this review was a determination that audit mechanisms are properly implemented to be reliable and tamper resistant, as well as that data that might be subject to tampering is properly validated and authenticated.
- Evaluate the risk that a user can 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 no issue was found.
- Evaluate the design and implementation to ensure that sound, generally accepted engineering practices are followed, checking to verify that code is defensively written against:
  - Bad data
  - Errors in other modules
  - Changes in environment
  - User errors
  - 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 no issue was found.

- Evaluate 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 no issue was found.
- Evaluate the code 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.

Software code requirements were found to be at issue within the **VSAP 3.0** source code base reviewed, as noted in this section. As a result, discrepancies were written against the code base.

Vulnerabilities

For any vulnerabilities discovered, SLI was tasked with identifying the particular standards applicable to each vulnerability.

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

- Voter: Usually has low knowledge of the voting technology design and configuration. Some may have more advanced knowledge. May carry out attacks designed by others.
- Elections official insider: Has a wide range of knowledge of the voting technology design and configuration. May have unrestricted access to voting technology for long periods of time. Their designated activities include:
  - o Set up and pre-election procedures;
  - o Election operation;
  - o Post-election processing of results; and
  - o Archiving and storage operations.

- Vendor insider: Has great knowledge of voting technology design and configuration. They have unlimited access to voting technology 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.

VSAP 3.0 Software Code Vulnerability Review

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 issues were found in the following areas:
  - Known language vulnerability, one instance was noted.

  Async code does not create new threads, but simply uses the current thread. Synchronous code will block the current thread, meaning that async code will potentially receive its response late, or not at all.
  - Dead code, one instance was noted.

  Commented out code was found in the source code base. Since it is a comment, it will not be built into the compiled version of the executable.

The source code was reviewed for evaluation 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.

VSAP Tally 2.0 Issues

This section reviews the one item noted from the VSAP Tally 2.0 examination conducted in 2019, in section "5.2 Static Code Analysis & Documentation Review," in the VSAP Tally 2.0 Software Test Report, that was scheduled to be addressed in this release, **VSAP 3.0**.

The item's text from the reviewed VSAP Tally 2.0 examination is *italicized* to differentiate its context with that of **VSAP 3.0**.

Item #25

Description: Calico container securityContext set to privileged = true.

securityContext: true is set for the container Calico, which controls network functions.

Assessment: The potential problem with this configuration is simply that the container is running effectively as root. An attacker could use this to reboot the system, delete files, modify passwords, etc.

However, there is a bug report filed at the following URL, which is attempting to deal with this issue related to Calico: https://github.com/projectc alico/calico/issues/2000

That said, it should be mentioned as a future improvement for the voting system, as this level of access to a machine via container is unnecessary and dangerous.

Developer Response: We agree that this is not an emergent finding but a future system version could see this remediated.

Severity: Low

SLI VSAP 3.0 Review: No update to this setting has been implemented. There does appear to be a fix for the issue with the v3.21.0 release of the Calico software. It appears to contain limitations and may not be feasible for VSAP at this time.

Determination: As a low-risk item, it is continuing to be monitored and will be addressed in a future version pending action from Calico, a third party.

Final Report

Findings were identified for the **VSAP 3.0** code base, as identified in the Review Results section above.

As directed by the California Secretary of State, this software testing report does not include any recommendation as to whether or not the system should be approved.

End of VSAP 3.0 Software Test Report