

# Security Analysis Report

Jan 18, 2025

by sec3 X-ray Auditor



# TABLE OF CONTENTS

Summary	3
Disclaimer	4
About sec3 X-ray Auditor	5
About sec3	5
Overview of the Result	6
Program: 678b3224655fbfca8fa92bf2_voltr-vault	7
Appendix A - sec3 Vulnerabilities and Exposures (SVE)	8

## Summary

sec3 X-ray Auditor ("sec3 Auditor") was used by VOLTR (the "Client") to conduct security analysis of a private local repository.

This analysis revealed 0 potential issues, of which 0 are critical.

This report presents the output from sec3 Auditor.

## Disclaimer

This report ("**Report**") includes the results of a security analysis, by **Sec3 X-ray Auditor**, of a specific build and/or version of the source code provided by the Client and specified in the Report ("**Assessed Code**").

The sole purpose of the Report is to provide the Client with the results of the security analysis of the Assessed Code. The Report does not apply to any other version and/or build of the Assessed Code.

Regardless of its contents, the Report does not (and shall not be interpreted to) provide any warranty, representation, or covenant that the Assessed Code: (i) is error and/or bug-free, (ii) has no security vulnerabilities, and/or (iii) does not infringe any third-party's rights. The Report is not, and shall not be construed or interpreted, in any manner, as, (i) an endorsement by the Company of the Assessed Code and/or of the Client, or (ii) investment advice or a recommendation to invest in the Assessed Code and/or the Client.

This Report shall be null and void if the Report (or any portion thereof) is altered in any manner.

## **About sec3 X-ray Auditor**

sec3 X-ray Auditor extracts essential code structure and relationships into a set of databases that enable sophisticated analysis of source code. Sec3 Software employs Maximal Concolic Execution (MCE) techniques, amongst others, to provide the ability to systematically explore code paths, encode path conditions and check path invariants.

At the time of this report, sec3 Auditor can scan 60 types of security vulnerabilities, including Missing Signer Check, Missing Owner Check, etc. Please refer to Appendix A for more information.

## **About sec3**

Founded by leading academics in the field of software security and senior industrial veterans, sec3 is a leading blockchain security company that currently focuses on Solana programs. We are also building sophisticated security tools that incorporate static analysis, penetration testing, and formal verification. At sec3, we identify and eliminate security vulnerabilities through the most rigorous process and aided by the most advanced analysis tools. For more information, check out our [website](#) and follow us on [twitter](#).

Congratulations! No issue was found in this audit.

## **Overview of the Result**

## **Program: 678b3224655fbfca8fa92bf2\_voltr-vault**

No issue was found in this program!

## Appendix A - sec3 Vulnerabilities and Exposures (SVE)

SVE	Checker	Description	Examples
<b>SVE10001</b>	ReentrancyEtherVulnerability	The function may suffer from reentrancy attacks due to the use of <code>call.value</code> , which can invoke an external contract's fallback function	<a href="#">Example</a>
<b>SVE10002</b>	ArbitrarySendERC20	The function may allow an attacker to send from an arbitrary address, instead of from the <code>msg.sender</code>	<a href="#">Example</a>
<b>SVE10003</b>	UnprotectedSelfDestruct	The function may allow an attacker to destruct the contract	<a href="#">Example</a>
<b>SVE10004</b>	MissingCalleeCheck	The function may be missing a check <code>callee != address(this)</code>	<a href="#">Example</a>
<b>SVE1001</b>	MissingSignerCheck	The account is missing signer check	<a href="#">Example</a>
<b>SVE1002</b>	MissingOwnerCheck	The account is missing owner check	<a href="#">Example</a>
<b>SVE1003</b>	IntegerAddOverflow	The add operation may result in overflows	<a href="#">Example</a>
<b>SVE1004</b>	IntegerUnderflow	The sub operation may result in underflows	<a href="#">Example</a>
<b>SVE1005</b>	IntegerMulOverflow	The mul operation may result in overflows	<a href="#">Example</a>
<b>SVE1006</b>	IntegerDivOverflow	The div operation may result in overflows	<a href="#">Example</a>
<b>SVE1007</b>	UnverifiedParsedAccount	The account is not validated before parsing its data	<a href="#">Example</a>



SVE	Checker	Description	Examples
<b>SVE1008</b>	DuplicateMutableAccount	These two accounts are both mutable and may be the same account	<a href="#">Example</a>
<b>SVE1009</b>	InsecureAccountClosing	The account may not be closed securely	<a href="#">Example</a>
<b>SVE1010</b>	TypeFullCosplay	These two account data types are fully compatible and can be used to launch type confusion attacks	<a href="#">Example</a>
<b>SVE1011</b>	TypePartialCosplay	These two account data types are partially compatible and may be exploited by type confusion attacks	<a href="#">Example</a>
<b>SVE1012</b>	DivideByZero	The arithmetic operation may result in a divide-by-zero error	<a href="#">Example</a>
<b>SVE1013</b>	AccountReInitialization	The account may be vulnerable to program re-initialization	<a href="#">Example</a>
<b>SVE1014</b>	BumpSeedNotValidated	The account's bump seed is not validated and may be vulnerable to seed canonicalization attacks	<a href="#">Example</a>
<b>SVE1015</b>	InsecurePDASharing	The PDA sharing with these seeds may be insecure	<a href="#">Example</a>
<b>SVE1016</b>	ArbitraryCPI	The CPI may be vulnerable and invoke an arbitrary program	<a href="#">Example</a>
<b>SVE1017</b>	MaliciousSimulation	The program may contain malicious simulation	<a href="#">Example</a>

SVE	Checker	Description	Examples
<b>SVE1018</b>	UnsafeSysVarAPI	The sysvar instructions API is unsafe and deprecated (wormhole exploit)	<a href="#">Example</a>
<b>SVE1019</b>	UnvalidatedAccount	The account may not be properly validated and may be untrustful	<a href="#">Example</a>
<b>SVE1020</b>	OutdatedDependency	The program has outdated and vulnerable dependencies	<a href="#">Example</a>
<b>SVE1021</b>	UnsafeRust	The program contains unsafe Rust code	<a href="#">Example</a>
<b>SVE1022</b>	OverPayment	The code misses checking to prevent over payment	<a href="#">Example</a>
<b>SVE1023</b>	StalePriceFeed	The code may use a stale price feed (solend loss)	<a href="#">Example</a>
<b>SVE1024</b>	MissInitTokenMint	The init instruction misses minting pool tokens	<a href="#">Example</a>
<b>SVE1025</b>	MissRentExempt	The account misses rent exempt check	<a href="#">Example</a>
<b>SVE1026</b>	MissFreezeAuthority	The account misses checking for freeze authority	<a href="#">Example</a>
<b>SVE1027</b>	FlashLoanRisk	The instruction may suffer from a flashloan risk due to internal price oracle	<a href="#">Example</a>
<b>SVE1028</b>	BidirectionalRounding	The arithmetics here may suffer from bidirectional rounding vulnerabilities	<a href="#">Example</a>
<b>SVE1029</b>	LossyCastTruncation	The cast operation here may lose precision due to truncation	<a href="#">Example</a>
<b>SVE1030</b>	UnvalidatedPDAAccount	The PDA account may not be properly validated and may be untrustful	<a href="#">Example</a>

SVE	Checker	Description	Examples
<b>SVE1031</b>	UnvalidatedDestinationAccount	The account is used as destination in token transfer without validation and it could be the same as the transfer source account	<a href="#">Example</a>
<b>SVE1032</b>	IncorrectAuthorityAccount	The PDA account may be incorrectly used as shared authority and may allow any account to transfer or burn tokens	<a href="#">Example</a>
<b>SVE1033</b>	InsecureAnchorInitIfNeeded	The `init_if_needed` keyword in anchor-lang prior to v0.24.x has a critical security bug	<a href="#">Example</a>
<b>SVE1034</b>	InsecureSPLTokenCPI	The spl_token account may be arbitrary prior to version v3.1.1	<a href="#">Example</a>
<b>SVE1035</b>	InsecureAssociatedTokenAccount	The associated token account is missing PDA key check and may be faked	<a href="#">Example</a>
<b>SVE1036</b>	InsecureAccountRealloc	The account realloc in solana_program prior to v1.10.29 may cause programs to malfunction	<a href="#">Example</a>
<b>SVE1037</b>	PDASeedCollisions	These two PDA accounts may have the same seeds, which may lead to PDA collisions	<a href="#">Example</a>
<b>SVE20001</b>	MissingInitAdminCheck	The init function misses checking admin uniqueness and may allow an attacker to call the init function more than once	<a href="#">Example</a>

SVE	Checker	Description	Examples
<b>SVE2002</b>	BitShiftOverflow	The bit shift operation may result in overflows	<a href="#">Example</a>
<b>SVE2003</b>	DivisionPrecisionLoss	The division operation here may lose precision	<a href="#">Example</a>
<b>SVE2004</b>	VulnerableI128Implementation	The I128 signed integer implementation in Move is not recommended and may be vulnerable. Consider using the built-in Move types only.	<a href="#">Example</a>
<b>SVE2001</b>	IncorrectLoopBreakLogic	Loop break instead of continue (jet-v1 exploit)	<a href="#">Example</a>
<b>SVE2002</b>	IncorrectConditionCheck	Liquidation condition $\geq$ should be $>$	<a href="#">Example</a>
<b>SVE2003</b>	ExponentialCalculation	The calculation has exponential complexity	<a href="#">Example</a>
<b>SVE2004</b>	IncorrectDivisionLogic	Incorrect checked_div instead of checked_ceil_div (spl-token-swap vulnerability: stable curve division)	<a href="#">Example</a>
<b>SVE2005</b>	IncorrectTokenCalculation	The token amount calculation may be incorrect. Consider using the reserves instead of the balances.	<a href="#">Example</a>
<b>SVE3001</b>	BestSecurityPractice	The code does not follow best security practices	<a href="#">Example</a>
<b>SVE3002</b>	CriticalUnusedCode	The code may be redundant or unused, but appears critical	<a href="#">Example</a>
<b>SVE3003</b>	InconsistentAnchor	The program uses Anchor inconsistently across different instructions	<a href="#">Example</a>

SVE	Checker	Description	Examples
<b>SVE3004</b>	InconsistentConfig	The configuration and initialization data are inconsistent	<a href="#">Example</a>
<b>SVE3005</b>	MissingCPIAccountReload	The token account's amount may be incorrect without calling reload after CPI	<a href="#">Example</a>
<b>SVE3006</b>	MissingUnstakeAccessControl	The unstake instruction may be missing an access_control account validation	<a href="#">Example</a>
<b>SVE3007</b>	OrderRaceCondition	The instruction may suffer from a race condition between order cancellation and order recreation by an attacker	<a href="#">Example</a>
<b>SVE3008</b>	NewAccountNotBackwardsCompatible	The account may break the ABI of the deployed on-chain program as it does not exist in the IDL available on Anchor	<a href="#">Example</a>
<b>SVE3009</b>	MutableAccountNotBackwardsCompatible	The mutable account may break the ABI of the deployed on-chain program as it is immutable according to the IDL available on Anchor	<a href="#">Example</a>
<b>SVE3010</b>	ReOrderAccountsNotBackwardsCompatible	These two accounts are reordered in the instruction and may break the ABI of the deployed on-chain program, according to the IDL available on Anchor	<a href="#">Example</a>

