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: 678b38f3655fbfca8fa92bf4_voltr-adaptor	7
Annendix A - sec3 Vulnerabilities and Exposures (SVF)	٩

SUMMARY

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

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 bugfree, (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.

OVERVIEW

Congratulations! No issue was found in this audit.

Overview of the Result

Program: 678b38f3655fbfca8fa92bf4_voltr-adaptor

Appendix A - sec3 Vulnerabilities and Exposures (SVE)

SVE	Checker	Description	Examples
		The function may suffer	
		from reentrancy attacks	
		due to the use of	
SVE10001	Reentrancy Ether Vulnerability	call.value, which can	<u>Example</u>
		invoke an external	
		contract's fallback	
		function	
		The function may allow an	
SVE10002	ArbitrarySendERC20	attacker to send from an	<u>Example</u>
30010002	Albitial y Selluchezo	arbitrary address, instead	<u>cxampte</u>
		of from the msg.sender	
		The function may allow an	
SVE10003	UnprotectedSelfDestruct	attacker to destruct the	<u>Example</u>
		contract	
	MissingCalleeCheck	The function may be	
SVE10004		missing a check callee !=	<u>Example</u>
		address(this)	
SVE1001	MissingSignerCheck	The account is missing	<u>Example</u>
	Managara Check	signer check	<u>CAUTIFICE</u>
SVE1002	MissingOwnerCheck	The account is missing	<u>Example</u>
	MISSINGOWNEI CHECK	owner check	<u>CAUTIFICE</u>
SVE1003	IntegerAddOverflow	The add operation may	<u>Example</u>
	integerAddoveritow	result in overflows	<u>cxampte</u>
SVE1004	IntegerUnderflow	The sub operation may	<u>Example</u>
3721004	integerondentow	result in underflows	cxampte
SVE100E	IntegerMulOverflow	The mul operation may	Evample
SVE1005	IIII.ERELIMITIONELLIOM	result in overflows	<u>Example</u>
SVE1006	06 IntegerDivOverflow	The div operation may	Evample
SVE1006		result in overflows	<u>Example</u>
		The account is not	
SVE1007	UnverifiedParsedAccount	validated before parsing	<u>Example</u>
		its data	

SVE	Checker	Description	Examples
SVE1008	DuplicateMutableAccount	These two accounts are	
		both mutable and may be	<u>Example</u>
		the same account	
EVE1000		The account may not be	
SVE1009	InsecureAccountClosing	closed securely	<u>Example</u>
		These two account data	
SVE1010	TypoFullCocplay	types are fully compatible	Evample
SVEIDIO	TypeFullCosplay	and can be used to launch	<u>Example</u>
		type confusion attacks	
		These two account data	
		types are partially	
SVE1011	TypePartialCosplay	compatible and may be	<u>Example</u>
		exploited by type	
		confusion attacks	
	DivideByZero	The arithmetic operation	
SVE1012		may result in a divide-by-	<u>Example</u>
-		zero error	
	AccountReInitialization	The account may be	
SVE1013		vulnerable to program re-	<u>Example</u>
		initialization	
		The account's bump seed	
SVE1014	BumpSeedNotValidated	is not validated and may	<u>Example</u>
312.01.	BumpJeedNotVallaatea	be vulnerable to seed	<u>exampte</u>
-		canonicalization attacks	
		The PDA sharing with	
SVE1015	InsecurePDASharing	these seeds may be	<u>Example</u>
		insecure	
		The CPI may be vulnerable	
SVE1016	ArbitraryCPI	and invoke an arbitrary	<u>Example</u>
		program	
SVE1017	MaliciousSimulation	The program may contain	<u>Example</u>
	เพลเเตเบนรวทานเสเเบท	malicious simulation	champte

SVE	Checker	Description	Examples
		The sysvar instructions	
CVC1010	Home Se Complete A DI	API is unsafe and	Current.
SVE1018	UnsafeSysVarAPI	deprecated (wormhole	<u>Example</u>
		exploit)	
		The account may not be	
SVE1019	UnvalidatedAccount	properly validated and	<u>Example</u>
		may be untrustful	
		The program has outdated	
SVE1020	OutdatedDependency	and vulnerable	<u>Example</u>
		dependencies	
SVE1021	UnsafeRust	The program contains	Evample
SVE1021	บทรสายหนรโ	unsafe Rust code	<u>Example</u>
EVE1022	OverPayment.	The code misses checking	Evansela
SVE1022	OverPayment	to prevent over payment	<u>Example</u>
SVE1023	StalePriceFeed	The code may use a stale	Evample
3VE1023	Staterficereeu	price feed (solend loss)	<u>Example</u>
EVE102/4	MisslaitTakanMint	The init instruction misses	Evample
SVE1024	MissInitTokenMint	minting pool tokens	<u>Example</u>
SVE1025	MissRentExempt	The account misses rent	<u>Example</u>
3VE1023	Misskentexempt	exempt check	cxampte
		The account misses	
SVE1026	MissFreezeAuthority	checking for freeze	<u>Example</u>
		authority	
		The instruction may suffer	
SVE1027	FlashLoanRisk	from a flashloan risk due	<u>Example</u>
		to internal price oracle	
		The arithmetics here may	
SVE1028	BidirectionalRounding	suffer from bidirectional	<u>Example</u>
		rounding vulnerabilities	
		The cast operation here	
SVE1029	LossyCastTruncation	may lose precision due to	<u>Example</u>
		truncation	
		The PDA account may not	
SVE1030	UnvalidatedPDAAccount	be properly validated and	<u>Example</u>
		may be untrustful	

SVE	Checker	Description	Examples
SVE1031		The account is used as	
		destination in token	
	UnvalidatedDestinationAccount	transfer without	<u>Example</u>
	OnvaluatedDestinationAccount	validation and it could be	<u>cxampte</u>
		the same as the transfer	
		source account	
		The PDA account may be	
		incorrectly used as shared	
SVE1032	IncorrectAuthorityAccount	authority and may allow	<u>Example</u>
		any account to transfer or	
		burn tokens	
		The `init_if_needed`	
SVE1033	InsecureAnchorInitIfNeeded	keyword in anchor-lang	<u>Example</u>
3451033	insecure Antino municeded	prior to v0.24.x has a	<u>cxampte</u>
		critical security bug	
		The spl_token account	
SVE1034	InsecureSPLTokenCPI	may be arbitrary prior to	<u>Example</u>
		version v3.1.1	
		The associated token	
SVE1035	InsecureAssociatedTokenAccount	account is missing PDA	<u>Example</u>
3461033	insecureassociateu i okenaccount	key check and may be	
		faked	
		The account realloc in	
SVE1036	InsecureAccountRealloc	solana_program prior to	<u>Example</u>
3021030	insecureAccountreattoc	v1.10.29 may cause	<u>exampte</u>
		programs to malfunction	
		These two PDA accounts	
SVE1037	PDASeedCollisions	may have the same seeds,	<u>Example</u>
3021037		which may lead to PDA	
		collisions	
SVE20001		The init function misses	
	MissingInitAdminCheck	checking admin	
		uniqueness and may allow	<u>Example</u>
		an attacker to call the init	
		function more than once	

SVEZOU02 BitShiftOverflow The bit shift operation may result in overflows SVEZOU03 DivisionPrecisionLoss The division operation here may lose precision here may lose lose precisi	SVE	Checker	Description	Examples
SVEZU003 DivisionPrecisionLoss The division operation here may lose precision bere may lose precision here may lose precision in Move is implementation in Move is not recommended and may be vulnerable. Consider using the built-in Move types only. SVEZU01 IncorrectLoopBreakLogic Loop break instead of continue (jet-v1 exploit) Example SVEZU02 IncorrectConditionCheck Liquidation condition >= should be > SVEZU03 ExponentialCalculation The calculation has exponential complexity instead of checked_ceil_div_spl-token-swap vulnerability: stable curve division) SVEZU04 IncorrectDivisionLogic checked_ceil_div_spl-token-swap vulnerability: stable curve division) The token amount calculation may be incorrect. Consider using the reserves instead of the balances. SVEZU05 IncorrectTokenCalculation The code does not follow best security practices SVEZU06 Example SVEZU07 ExiticalUnusedCode The code may be redundant or unused, but appears critical The program uses Anchor inconsistently across Example	SVEZOOOZ	BitShiftOverflow	The bit shift operation	Cymente
SVE2003 DivisionPrecisionLoss here may lose precision The I128 signed integer implementation in Move is not recommended and may be vulnerable. Consider using the built-in Move types only. SVE2001 IncorrectLoopBreakLogic Loop break instead of continue (jet-v1 exploit) SVE2002 IncorrectConditionCheck Should be > SVE2003 ExponentialCalculation Example SVE2004 IncorrectDivisionLogic The calculation has exponential complexity instead of checked_ceil_div (spl-token-swap vulnerability: stable curve division) SVE2005 IncorrectTokenCalculation incorrect. Consider using the reserves instead of the balances. SVE2005 CriticalUnusedCode The code does not follow best security practices The code may be redundant or unused, but appears critical The program uses Anchor Incorrest Example The program uses Anchor IncorrestSyample Example Example Example Example Example Example Example Example Example Example Example Example Example Example Example			may result in overflows	<u>cxampte</u>
SVEZOU24 VulnerableI12BImplementation The I12B signed integer implementation in Move is not recommended and may be vulnerable. Consider using the built-in Move types only. Loop break instead of continue (jet-v1 exploit) SVEZOU2 IncorrectConditionCheck SVEZOU3 ExponentialCalculation Example SVEZOU3 ExponentialCalculation The calculation has exponential complexity instead of checked_ceil_div (spl-token-swap vulnerability: stable curve division) SVEZOU4 IncorrectTokenCalculation Fine token amount calculation may be incorrect. Consider using the reserves instead of the balances. SVEZOU5 Example SVEZOU5 Example SVEZOU5 Example SVEZOU5 IncorrectTokenCalculation BestSecurityPractice The code does not follow best security practices The code does not follow best security practices The code may be redundant or unused, but appears critical The program uses Anchor The program uses Anchor inconsistently across Example	CNESOUS	DivisionProcision	The division operation	Evample
SVEZO004 VulnerableI128Implementation in Move is not recommended and may be vulnerable. Consider using the built-in Move types only. SVEZO01 IncorrectLoopBreakLogic Loop break instead of continue (jet-v1 exploit) Example SVEZO02 IncorrectConditionCheck Liquidation condition >= should be > SVEZO03 ExponentialCalculation The calculation has exponential complexity instead of checked_ceil_div (spl-token-swap vulnerability: stable curve division) SVEZO04 IncorrectTokenCalculation The token amount calculation may be incorrect. Consider using the reserves instead of the balances. SVEZO05 BestSecurityPractice The code does not follow best security practices SVEZO06 CriticalUnusedCode The code may be redundant or unused, but appears critical The program uses Anchor inconsistently across Example SVEZO07 IncorrectTokenCalculation The program uses Anchor inconsistently across Example	SVE20003	DIVISIONETECISIONCOSS	here may lose precision	cxampte
SVE2004 Vulnerable 28 mplementation			The I128 signed integer	
SVE2004 VulnerableI128Implementation may be vulnerable. Consider using the built-in Move types only. Loop break instead of continue (jet-v1 exploit) SVE2002 IncorrectConditionCheck SVE2003 ExponentialCalculation SVE2004 IncorrectDivisionLogic SVE2004 IncorrectDivisionLogic SVE2004 IncorrectDivisionLogic SVE2005 IncorrectTokenCalculation SVE2005 IncorrectTokenCalculation SVE2005 IncorrectTokenCalculation SVE2006 IncorrectTokenCalculation SVE2006 IncorrectTokenCalculation SVE2007 IncorrectTokenCalculation SVE2008 IncorrectTokenCalculation SVE2009 IncorrectTokenCalculation SVE2009 IncorrectTokenCalculation The token amount calculation may be incorrect. Consider using the reserves instead of the balances. SVE3001 BestSecurityPractice SVE3001 The code does not follow best security practices The code may be redundant or unused, but appears critical The program uses Anchor inconsistently across Example			implementation in Move is	
SVEZOO1 IncorrectConditionCheck Consider using the built-in Move types only. Loop break instead of continue (jet-v1 exploit) SVEZOO2 IncorrectConditionCheck Liquidation condition >= Sxample should be > SVEZOO3 ExponentialCalculation Example SVEZOO4 IncorrectDivisionLogic Checked_ceil_div (spl-token-swap vulnerability: stable curve division) SVEZOO5 IncorrectTokenCalculation incorrect. Consider using the reserves instead of the balances. SVEZOO5 Example SVEZOO5 CriticalUnusedCode The code does not follow best security practices The code may be redundant or unused, but appears critical The program uses Anchor inconsistently across Example	SVEZOOO/i	Vulnorable 1781 mplomentation	not recommended and	Evamplo
SVE2001 IncorrectLoopBreakLogic Loop break instead of continue (jet-v1 exploit) SVE2002 IncorrectConditionCheck Liquidation condition >= Example SVE2003 ExponentialCalculation	30020004	vutnerabtenzomptementation	may be vulnerable.	cxampte
SVE2001 IncorrectLoopBreakLogic Continue (jet-v1 exploit) SVE2002 IncorrectConditionCheck Liquidation condition >= should be > SVE2003 ExponentialCalculation The calculation has exponential complexity Incorrect checked_div instead of checked_ceit_div (spl-token-swap vulnerability: stable curve division) SVE2004 IncorrectTokenCalculation incorrect. Consider using the reserves instead of the balances. SVE2005 BestSecurityPractice The code does not follow best security practices SVE3001 BestSecurityPractice The code may be redundant or unused, but appears critical The program uses Anchor inconsistently across Example			Consider using the built-in	
SVEZOO2 IncorrectLoopBreakLogic continue (jet-v1 exploit) SVEZOO2 IncorrectConditionCheck			Move types only.	
SVE2002 IncorrectConditionCheck Liquidation condition >= should be > SVE2003 ExponentialCalculation Example SVE2004 IncorrectDivisionLogic Incorrect checked_div instead of checked_ceil_div (spl-token-swap vulnerability: stable curve division) SVE2005 IncorrectTokenCalculation incorrect. Consider using the reserves instead of the balances. SVE3001 BestSecurityPractice The code does not follow best security practices The code may be redundant or unused, but appears critical appears critical The program uses Anchor inconsistently across Example	EVE2001	Incompatition Dynamic	Loop break instead of	Cummin
SVE2002 IncorrectConditionCheck should be > SVE2003 ExponentialCalculation Example SVE2004 ExponentialCalculation Example SVE2004 IncorrectDivisionLogic Checked_ceil_div (spl- token-swap vulnerability: stable curve division) The token amount calculation may be incorrect. Consider using the reserves instead of the balances. SVE3001 BestSecurityPractice The code does not follow best security practices The code may be redundant or unused, but appears critical The program uses Anchor SVE3003 InconsistentAnchor inconsistently across Example		псотгестсоорвгеаксовіс	continue (jet-v1 exploit)	
SVEZOO3 ExponentialCalculation ExponentialCalculation ExponentialCalculation Example exponential complexity Incorrect checked_div instead of checked_ceil_div (spl- token-swap vulnerability: stable curve division) The token amount calculation may be incorrect. Consider using the reserves instead of the balances. SVE3001 BestSecurityPractice The code does not follow best security practices The code may be redundant or unused, but example appears critical The program uses Anchor SVE3003 InconsistentAnchor Inconsistently across Example	EVE2002	lane was at Condition Classic	Liquidation condition >=	Cymraela
SVE2003 ExponentialCalculation exponential complexity Incorrect checked_div instead of checked_ceil_div (spl-token-swap vulnerability: stable curve division) SVE2004 IncorrectTokenCalculation incorrect. Consider using the reserves instead of the balances. SVE3001 BestSecurityPractice The code does not follow best security practices SVE3002 CriticalUnusedCode redundant or unused, but example appears critical appears critical The program uses Anchor inconsistently across Example	SVE2002	IncorrectConditionCneck	should be >	<u>Example</u>
SVE2004 IncorrectDivisionLogic checked_ceil_div (spl- token-swap vulnerability: stable curve division) The token amount calculation may be incorrect. Consider using the reserves instead of the balances. SVE3001 BestSecurityPractice The code does not follow best security practices The code may be redundant or unused, but example appears critical The program uses Anchor SVE3003 InconsistentAnchor Example The program uses Anchor inconsistently across Example	EVEZOO2	Company antical College Atlanta	The calculation has	C
SVE2004 IncorrectDivisionLogic checked_ceil_div (spl- token-swap vulnerability: stable curve division) The token amount calculation may be incorrect. Consider using the reserves instead of the balances. SVE3001 BestSecurityPractice The code does not follow best security practices The code may be redundant or unused, but appears critical The program uses Anchor SVE3003 InconsistentAnchor inconsistently across Example	SVE2003	ExponentialCalculation	exponential complexity	<u>Example</u>
SVE2004 IncorrectDivisionLogic checked_ceil_div (spl- token-swap vulnerability: stable curve division) The token amount calculation may be incorrect. Consider using the reserves instead of the balances. SVE3001 BestSecurityPractice The code does not follow best security practices The code may be redundant or unused, but appears critical The program uses Anchor SVE3003 InconsistentAnchor inconsistently across Example			Incorrect checked_div	
token-swap vulnerability: stable curve division) The token amount calculation may be incorrect. Consider using the reserves instead of the balances. SVE3001 BestSecurityPractice The code does not follow best security practices The code may be redundant or unused, but appears critical The program uses Anchor svE3003 InconsistentAnchor inconsistently across Example			instead of	
SVE2005 IncorrectTokenCalculation Example SVE2005 IncorrectTokenCalculation SVE3001 BestSecurityPractice The code does not follow best security practices The code may be redundant or unused, but appears critical The program uses Anchor inconsistently across Example	SVE2004	IncorrectDivisionLogic	checked_ceil_div (spl-	<u>Example</u>
The token amount calculation may be incorrect. Consider using the reserves instead of the balances. The code does not follow best security practices The code may be redundant or unused, but appears critical The program uses Anchor svE3003 InconsistentAnchor The token amount calculation may be incorrect. Consider using Example Example The code does not follow best security practices The program uses Anchor inconsistently across Example			token-swap vulnerability:	
SVE2005 IncorrectTokenCalculation incorrect. Consider using the reserves instead of the balances. SVE3001 BestSecurityPractice The code does not follow best security practices The code may be redundant or unused, but appears critical The program uses Anchor inconsistently across Example			stable curve division)	
SVE2005 IncorrectTokenCalculation incorrect. Consider using the reserves instead of the balances. SVE3001 BestSecurityPractice The code does not follow best security practices The code may be redundant or unused, but appears critical The program uses Anchor inconsistently across Example			The token amount	
the reserves instead of the balances. SVE3001 BestSecurityPractice The code does not follow best security practices The code may be The code may be redundant or unused, but appears critical The program uses Anchor inconsistently across Example			calculation may be	
the balances. The code does not follow best security practices The code may be The code may be The code may be redundant or unused, but example appears critical The program uses Anchor SVE3003 InconsistentAnchor inconsistently across Example	SVE2005	$Incorrect Token {\tt Calculation}$	incorrect. Consider using	<u>Example</u>
SVE3001 BestSecurityPractice The code does not follow best security practices The code may be The code may be redundant or unused, but appears critical The program uses Anchor SVE3003 InconsistentAnchor The program uses Anchor inconsistently across Example			the reserves instead of	
SVE3001 BestSecurityPractice The code may be SVE3002 CriticalUnusedCode redundant or unused, but Example appears critical The program uses Anchor SVE3003 InconsistentAnchor inconsistently across Example			the balances.	
best security practices The code may be SVE3002 CriticalUnusedCode redundant or unused, but Example appears critical The program uses Anchor SVE3003 InconsistentAnchor inconsistently across Example	SVE3001	RostSocurity/Practice	The code does not follow	Evample
SVE3002 CriticalUnusedCode redundant or unused, but Example appears critical The program uses Anchor SVE3003 InconsistentAnchor inconsistently across Example		Desibecultyridetile	best security practices	<u> </u>
appears critical The program uses Anchor SVE3003 InconsistentAnchor inconsistently across Example			The code may be	
The program uses Anchor SVE3003 InconsistentAnchor inconsistently across Example	SVE3002	CriticalUnusedCode	redundant or unused, but	<u>Example</u>
SVE3003 InconsistentAnchor inconsistently across <u>Example</u>			appears critical	
			The program uses Anchor	
different instructions	SVE3003	InconsistentAnchor	inconsistently across	<u>Example</u>
			different instructions	

SVE	Checker	Description	Examples
		The configuration and	
SVE3004	InconsistentConfig	initialization data are	<u>Example</u>
		inconsistent	
		The token account's	
SVE3005	MissingCDIAssoumtDolood	amount may be incorrect	Evample
3453003	MissingCPIAccountReload	without calling reload	<u>Example</u>
		after CPI	
		The unstake instruction	
SVE3006	MissingUnstakeAccessControl	may be missing an	<u>Example</u>
3453000	MISSINGUISTAREACCESSCOILLIOL	access_control account	<u>cxampte</u>
		validation	
		The instruction may suffer	
		from a race condition	
SVE3007	OrderRaceCondition	between order	<u>Example</u>
		cancellation and order	
		recreation by an attacker	
		The account may break	
		the ABI of the deployed	
SVE3008	New Account Not Backwards Compatible	on-chain program as it	<u>Example</u>
		does not exist in the IDL	
		available on Anchor	
		The mutable account may	
		break the ABI of the	
		deployed on-chain	
SVE3009	${\it Mutable Account Not Backwards Compatible}$	program as it is	<u>Example</u>
		immutable according to	
		the IDL available on	
		Anchor	
		These two accounts are	
		reordered in the	
		instruction and may break	
SVE3010	ReOrder Accounts Not Backwards Compatible	the ABI of the deployed	<u>Example</u>
		on-chain program,	
		according to the IDL	
		available on Anchor	