



Sovryn
Zero Protocol
SMART CONTRACT AUDIT
13.10.2021

Made in Germany by Chainsulting.de



Table of contents

1. Disclaimer	3
2. About the Project and Company	4
2.1 Project Overview	5
3. Vulnerability & Risk Level.....	6
4. Auditing Strategy and Techniques Applied	7
4.1 Methodology.....	7
4.2 Used Code from other Frameworks/Smart Contracts	8
4.3 Tested Contract Files.....	9
4.4 Metrics / CallGraph	12
4.5 Metrics / Source Lines & Risk	13
4.6 Metrics / Capabilities.....	14
4.7 Metrics / Source Unites in Scope.....	15
5. Scope of Work.....	22
5.1 Manual and Automated Vulnerability Test	23
5.1.1 Wrong import of OpenZeppelin library.....	23
5.2 Verify claims	24
6. Executive Summary	25
7. Deployed Smart Contract.....	25

1. Disclaimer

The audit makes no statements or warranties about utility of the code, safety of the code, suitability of the business model, investment advice, endorsement of the platform or its products, regulatory regime for the business model, or any other statements about fitness of the contracts to purpose, or their bug free status. The audit documentation is for discussion purposes only.

The information presented in this report is confidential and privileged. If you are reading this report, you agree to keep it confidential, not to copy, disclose or disseminate without the agreement of 1A1Z Limited. If you are not the intended receptor of this document, remember that any disclosure, copying or dissemination of it is forbidden.

Major Versions / Date	Description
0.1 (28.09.2021)	Layout
0.2 (30.09.2021)	Test Deployment
0.5 (01.10.2021)	Automated Security Testing Manual Security Testing
0.6 (03.10.2021)	Testing SWC Checks
0.7 (06.10.2021)	Verify Claims
0.9 (10.10.2021)	Summary and Recommendation
1.0 (13.10.2021)	Final document
1.1 (TBA)	Added deployed contract addresses

2. About the Project and Company

Company address:

1A1Z Limited
64 Southwark Bridge Road
London SE1 0AS
United Kingdom

Website: <https://www.sovryn.app>

Twitter: <https://twitter.com/SovrynBTC>

Medium: <https://medium.com/sovryn>

Telegram: <https://t.me/SovrynBitcoin>

Discord: <https://discord.gg/J22WS6z>

Github: <https://github.com/DistributedCollective>



2.1 Project Overview

Zero is a decentralized protocol based on Liquidity that allows Bitcoin holders to obtain maximum liquidity against their collateral without paying interest. After locking up rBTC as collateral in a smart contract and creating an individual position called a "line of credit (trove)", the user can get instant liquidity by minting ZUSD, a USD-pegged stablecoin.

Each line of credit is required to be collateralized at a minimum of 110%. Any owner of ZUSD can redeem their stablecoins for the underlying collateral at any time. The redemption mechanism along with algorithmically adjusted fees guarantee a minimum stablecoin value of USD 1.

The stablecoin tokens are economically geared towards maintaining value of 1 ZUSD = \$1 USD, due to the following properties:

1. The system is designed to always be over-collateralized - the dollar value of the locked rBTC exceeds the dollar value of the issued stablecoins
2. The stablecoins are fully redeemable - users can always swap \$x worth of ZUSD for \$x worth of rBTC (minus fees), directly with the system.
3. The system algorithmically controls the generation of ZUSD through a variable issuance fee.

An unprecedented liquidation mechanism based on incentivized stability deposits and a redistribution cycle from riskier to safer lines of credit provides stability at a much lower collateral ratio than current systems. Stability is maintained via economically-driven user interactions and arbitrage, rather than by active governance or monetary interventions.

The protocol has built-in incentives that encourage both early adoption and the operation of multiple front ends, enhancing decentralization.

3. Vulnerability & Risk Level

Risk represents the probability that a certain source-threat will exploit vulnerability, and the impact of that event on the organization or system. Risk Level is computed based on CVSS version 3.0.

Level	Value	Vulnerability	Risk (Required Action)
Critical	9 – 10	A vulnerability that can disrupt the contract functioning in a number of scenarios, or creates a risk that the contract may be broken.	Immediate action to reduce risk level.
High	7 – 8.9	A vulnerability that affects the desired outcome when using a contract, or provides the opportunity to use a contract in an unintended way.	Implementation of corrective actions as soon as possible.
Medium	4 – 6.9	A vulnerability that could affect the desired outcome of executing the contract in a specific scenario.	Implementation of corrective actions in a certain period.
Low	2 – 3.9	A vulnerability that does not have a significant impact on possible scenarios for the use of the contract and is probably subjective.	Implementation of certain corrective actions or accepting the risk.
Informational	0 – 1.9	A vulnerability that have informational character but is not effecting any of the code.	An observation that does not determine a level of risk

4. Auditing Strategy and Techniques Applied

Throughout the review process, care was taken to evaluate the repository for security-related issues, code quality, and adherence to specification and best practices. To do so, reviewed line-by-line by our team of expert pentesters and smart contract developers, documenting any issues as there were discovered.

4.1 Methodology

The auditing process follows a routine series of steps:

1. Code review that includes the following:
 - i. Review of the specifications, sources, and instructions provided to Chainsulting to make sure we understand the size, scope, and functionality of the smart contract.
 - ii. Manual review of code, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
 - iii. Comparison to specification, which is the process of checking whether the code does what the specifications, sources, and instructions provided to Chainsulting describe.
2. Testing and automated analysis that includes the following:
 - i. Test coverage analysis, which is the process of determining whether the test cases are actually covering the code and how much code is exercised when we run those test cases.
 - ii. Symbolic execution, which is analysing a program to determine what inputs causes each part of a program to execute.
3. Best practices review, which is a review of the smart contracts to improve efficiency, effectiveness, clarify, maintainability, security, and control based on the established industry and academic practices, recommendations, and research.
4. Specific, itemized, actionable recommendations to help you take steps to secure your smart contracts.



4.2 Used Code from other Frameworks/Smart Contracts (direct imports)

Dependency / Import Path	Source
IERC20.sol	https://github.com/OpenZeppelin/openzeppelin-contracts/blob/v3.0.0/contracts/token/ERC20/IERC20.sol
Ownable.sol	https://github.com/OpenZeppelin/openzeppelin-contracts/blob/v3.0.0/contracts/access/Ownable.sol
SafeMath.sol	https://github.com/OpenZeppelin/openzeppelin-contracts/blob/v3.0.0/contracts/math/SafeMath.sol
IERC2612.sol	https://github.com/OpenZeppelin/openzeppelin-contracts/pull/2237/
Initializable.sol	https://github.com/OpenZeppelin/openzeppelin-upgrades/blob/master/packages/core/contracts/Initializable.sol

4.3 Tested Contract Files

The following are the MD5 hashes of the reviewed files. A file with a different MD5 hash has been modified, intentionally or otherwise, after the security review. You are cautioned that a different MD5 hash could be (but is not necessarily) an indication of a changed condition or potential vulnerability that was not within the scope of the review

File	Fingerprint (MD5)
./contracts/ZUSDTToken.sol	ba16824559fa1dfd5efc0a50695f1e04
./contracts/LiquidityBaseParams.sol	353a090d48b5626af09b45d4c09b9a9f
./contracts/HintHelpers.sol	3e48058939cfa2c728b0a797ea1ee943
./contracts/SortedTrovesStorage.sol	3803e33e7e83943470f4c9bf3b7b8187
./contracts/SortedTroves.sol	af74e38c38152edae96a458437174b4e
./contracts/MultiTrovesGetter.sol	e16f75c9a8d7aa30a9faba472c6bc987
./contracts/DefaultPool.sol	fd953ffa666b426ac6ed2eda03fd763a
./contracts/CollSurplusPool.sol	fc3b29b8518060c0032f034bfdcf3d71
./contracts/Proxy/UpgradableProxy.sol	d456087c6c416934e4b59a27b34d5a7c
./contracts/StabilityPoolStorage.sol	0c003d0190f7b53b17b1f710ebb34453
./contracts/PriceFeedStorage.sol	0ba4d1964f31d0fa85d30df347b23692
./contracts/ActivePoolStorage.sol	4ba9695f6509daf620b38c828308a36c
./contracts/PriceFeed.sol	063a81b877d03af5ec63c11c42e3dbfa
./contracts/BorrowerOperationsStorage.sol	955a5965782b61d615c6c95974ae1bb5
./contracts/TroveManager.sol	f478ef4872c0e21b7e84948770e97a63
./contracts/BorrowerOperations.sol	4bd9c358915b39e548bef2ddf682faba
./contracts/TroveManagerStorage.sol	486717cc1f7733b6477ed6c0376c0209
./contracts/Proxy/ETHTransferScript.sol	2490690b25902621b2f4c3ca27e2e2c1
./contracts/Proxy/BorrowerWrappersScript.sol	952029dd578a8700ca3dc56693dd9117
./contracts/Proxy/Proxy.sol	d0519169277608adab5fe60a0260427e
./contracts/ActivePool.sol	8c9b58b384991fb86486e126af8b196e
./contracts/Proxy/ZEROStakingScript.sol	f08c03ad44eb1a0c5ff739e004389fd0
./contracts/Proxy/TokenScript.sol	c9452bb8edce6fa6aa237ef799cdd4d1

./contracts/Proxy/TroveManagerScript.sol	8968f8513f75fcd40a1c292990336d79
./contracts/Proxy/StabilityPoolScript.sol	d8e1d73d2a588a7a44518e6efebd88f2
./contracts/Proxy/BorrowerOperationsScript.sol	1dc0de487cab1a33a7fdca940b37e37d
./contracts/Interfaces/ISortedTrove.sol	05987e08cee35420d1f2f5913c560619
./contracts/Interfaces/IBorrowerOperations.sol	85ddc6145ba723a6f65c53dd0eb896
./contracts/MultiTroveGetterStorage.sol	773572cf15b7d9b1b9ea5deb5afe8c82
./contracts/StabilityPool.sol	3b0ba8d8dfc0ec39d8d4d8d8d86423a4
./contracts/GasPool.sol	f653197914b58fcc9bab2e9d1b2e1ce2
./contracts/HintHelpersStorage.sol	34befcd8567c291310cae07f790d200b
./contracts/DefaultPoolStorage.sol	f94e2eb1657985f1a6a62a1c2aa48101
./contracts/ZUSDTokenStorage.sol	ba16824559fa1dfd5efc0a50695f1e04
./contracts/CollSurplusPoolStorage.sol	f08054d810d405ade4ed920708ef8013
./contracts/Interfaces/IStabilityPool.sol	e1284a20ff1eb291959242f22183d8b1
./contracts/Interfaces/IDefaultPool.sol	1fb03bffdcdcfefa00de35922efff644c
./contracts/Interfaces/ITroveManager.sol	06c2bf20ab72b49e8450f0a123084c63
./contracts/ZERO/CommunityIssuance.sol	059dd9bcb75837958f9bf474e1741055
./contracts/ZERO/LockupContractFactory.sol	43708d5cbcd95d682b87456cb33ea358
./contracts/Interfaces/IZEROToken.sol	685cd1fff18bfabf8cdab637e9f65ad7
./contracts/ZERO/CommunityIssuanceBase.sol	190f14eada9b19ab44c7e5a90f9147f8
./contracts/Interfaces/ILiquidityBaseParams.sol	6aec9f4cb8f33a713a11dde120f86143
./contracts/Interfaces/IPriceFeed.sol	07c792dfbda9a25ba0d0367af4854da4
./contracts/ZERO/CommunityIssuanceStorage.sol	f2f5dd803ec33c409ea293da49897628
./contracts/Interfaces/IPool.sol	700fe7447542a0e3cfbf4df3cfe806c9
./contracts/ZERO/LockupContract.sol	93933c84ccacc6e8a9515bf6d738a5c4
./contracts/Interfaces/ICommunityIssuance.sol	5e9155e20648394dc0fe2e42e1ace484
./contracts/ZERO/ZEROToken.sol	033a327b80361bd2a2f98f1ea6557be5
./contracts/Interfaces/IActivePool.sol	a378573f606df0303cdf482bf32d87a7
./contracts/ZERO/ZEROSTakingStorage.sol	4a8c44c244c2c3c355985ebb059e9f83
./contracts/ZERO/LockupContractFactoryStorage.sol	22e9db5ed8f0463bc62495988fc2c1f6
./contracts/Interfaces/IZUSDToken.sol	cd27af9c68e8373fc508fba0dfb2d1c4
./contracts/Interfaces/ILockupContractFactory.sol	ba99feea14e516b8624e1d534ac7a43d

./contracts/ZERO/ZEROStaking.sol	f20a1cf932e4e40c750273a7ad429e94
./contracts/Interfaces/IZEROStaking.sol	99140402b3d89af3a7ba060290c53360
./contracts/Interfaces/ICollSurplusPool.sol	052c09c583db8331eb9dba67df833410
./contracts/ZERO/SovStakersIssuance.sol	28f6a78fc7f23545a2128fddaa0decc4
./contracts/ZERO/ZEROTokenStorage.sol	bfaf91db076b3beb54ddfc7f2188cd92
./contracts/Interfaces/ILiquidityBase.sol	25a5b012ecce103271708d76f2701ae7

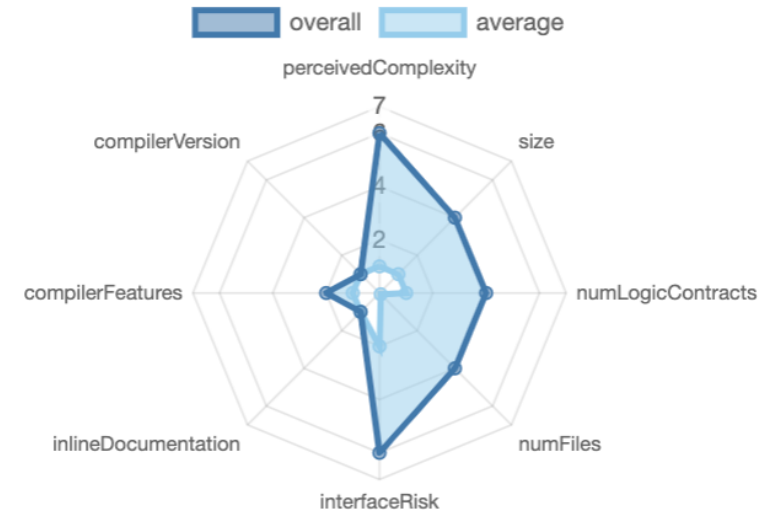
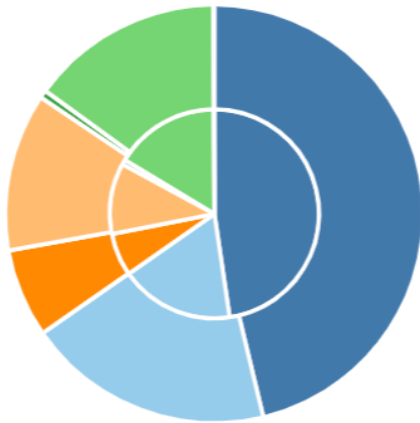
4.4 Metrics / CallGraph



View full version: <http://chainsulting.de/wp-content/uploads/2021/10/solidity-metrics-zero.html>


4.5 Metrics / Source Lines & Risk

source comment single block mixed
empty todo blockEmpty





4.6 Metrics / Capabilities

Solidity Versions observed		 Experimental Features	 Can Receive Funds	 Uses Assembly	 Has Destroyable Contracts
0.6.11		ABIEncoderV2	yes	yes (5 asm blocks)	


 Transfers ETH	 Low-Level Calls	 DelegateCall	 Uses Hash Functions	 ECRecover	 New/Create/Create2
yes		yes	yes	yes	yes → NewContract:LockupContract

Exposed Functions















This section lists functions that are explicitly declared public or payable. Please note that getter methods for public stateVars are not included.














 Public	 Payable				
328	19				
External	Internal	Private	Pure	View	
285	354	4	23	210	
















StateVariables













Total	 Public
173	114















4.7 Metrics / Source Unites in Scope













Type	File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
	packages/contracts/contracts/ZUSDToken.sol	1	_____	271	255	177	28	133	
	packages/contracts/contracts/LiquidityBaseParams.sol	1	_____	65	65	41	13	36	_____
	packages/contracts/contracts/HintHelpers.sol	1	_____	170	147	82	30	73	
	packages/contracts/contracts/SortedTroveStorage.sol	1	_____	37	37	25	10	7	_____
	packages/contracts/contracts/SortedTroveStorage.sol	1	_____	396	396	195	147	137	_____
	packages/contracts/contracts/MultiTroveGetter.sol	1	_____	118	112	86	6	80	
	packages/contracts/contracts/DefaultPool.sol	1	_____	103	97	57	19	46	
	packages/contracts/contracts/CollSurplusPool.sol	1	_____	111	103	68	8	44	
	packages/contracts/contracts/Proxy/UpgradableProxy.sol	1	_____	36	36	7	26	7	_____





Type	File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
	packages/contracts/contracts/StabilityPoolStorage.sol	1		102	102	49	35	27	
	packages/contracts/contracts/PriceFeedStorage.sol	1		17	17	10	2	8	
	packages/contracts/contracts/ActivePoolStorage.sol	1		20	20	11	7	10	
	packages/contracts/contracts/PriceFeed.sol	1		65	65	38	15	33	
	packages/contracts/contracts/BorrowerOperationsStorage.sol	1		35	35	20	3	12	
	packages/contracts/contracts/TroveManager.sol	1		1152	1022	746	119	408	
	packages/contracts/contracts/BorrowerOperations.sol	1		653	551	381	65	251	
	packages/contracts/contracts/TroveManagerStorage.sol	1		96	96	49	20	27	
	packages/contracts/contracts/Proxy/ETHTransferScript.sol	1		11	11	7	1	4	
	packages/contracts/contracts/Proxy/BorrowerWrappersScript.sol	1		160	160	115	12	102	

Type	File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
	packages/contracts/contracts/Proxy/Proxy.sol	1		104	104	47	47	87	
	packages/contracts/contracts/ActivePool.sol	1		134	129	78	30	51	
	packages/contracts/contracts/Proxy/ZERO StakingScript.sol	1		20	20	13	1	11	
	packages/contracts/contracts/Proxy/Token Script.sol	1		42	42	29	1	27	
	packages/contracts/contracts/Proxy/Trove ManagerScript.sol	1		38	30	22	1	12	
	packages/contracts/contracts/Proxy/StabilityPoolScript.sol	1		30	30	20	1	18	
	packages/contracts/contracts/Proxy/BorrowerOperationsScript.sol	1		48	48	34	1	41	
	packages/contracts/contracts/Interfaces/ISortedTrove.sol		1	116	24	7	74	31	
	packages/contracts/contracts/Interfaces/IBorrowerOperations.sol		1	136	42	16	77	37	
	packages/contracts/contracts/MultiTroveGetterStorage.sol	1		13	13	8	2	5	

Type	File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
	packages/contracts/contracts/StabilityPool.sol	1	_____	916	889	429	308	281	
	packages/contracts/contracts/GasPool.sol	1	_____	18	18	3	12	1	
	packages/contracts/contracts/HintHelpersStorage.sol	1	_____	15	15	9	1	6	_____
	packages/contracts/contracts/DefaultPoolStorage.sol	1	_____	15	15	10	3	8	_____
	packages/contracts/contracts/ZUSDTokenStorage.sol	1	_____	40	40	23	8	20	_____
	packages/contracts/contracts/CollSurplusPoolStorage.sol	1	_____	18	18	10	3	9	_____
	packages/contracts/contracts/Interfaces/IStabilityPool.sol	_____	1	220	85	28	146	27	_____
	packages/contracts/contracts/Interfaces/IDefaultPool.sol	_____	1	18	17	7	5	5	_____
	packages/contracts/contracts/Interfaces/ITroveManager.sol	_____	1	280	61	32	134	83	_____
	packages/contracts/contracts/ZERO/CommunityIssuance.sol	1	_____	23	23	13	2	6	_____

Type	File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
	packages/contracts/contracts/ZERO/LockupContractFactory.sol	1		65	65	34	17	34	
	packages/contracts/contracts/Interfaces/IZEROToken.sol		1	26	21	8	7	9	
	packages/contracts/contracts/ZERO/CommunityIssuanceBase.sol	1		97	89	48	17	39	
	packages/contracts/contracts/Interfaces/ILiquidityBaseParams.sol		1	25	8	3	7	13	
	packages/contracts/contracts/Interfaces/IPriceFeed.sol		1	17	16	4	8	3	
	packages/contracts/contracts/ZERO/CommunityIssuanceStorage.sol	1		44	44	13	20	11	
	packages/contracts/contracts/Interfaces/IPool.sol		1	32	20	9	11	9	
	packages/contracts/contracts/ZERO/LockupContract.sol	1		86	86	45	21	24	
	packages/contracts/contracts/Interfaces/ICommunityIssuance.sol		1	35	21	6	14	7	
	packages/contracts/contracts/ZERO/ZERO Token.sol	1		324	305	187	56	152	

Type	File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
	packages/contracts/contracts/Interfaces/IActivePool.sol	_____	1	27	26	8	13	5	_____
	packages/contracts/contracts/ZERO/ZERO StakingStorage.sol	1	_____	34	34	21	5	14	_____
	packages/contracts/contracts/ZERO/LockupContractFactoryStorage.sol	1	_____	16	16	8	2	7	_____
	packages/contracts/contracts/Interfaces/IZUSDTToken.sol	_____	1	27	20	9	3	13	_____
	packages/contracts/contracts/Interfaces/ILockupContractFactory.sol	_____	1	29	20	10	9	7	_____
	packages/contracts/contracts/ZERO/ZERO Staking.sol	1	_____	224	213	143	18	99	
	packages/contracts/contracts/Interfaces/IZEROSTaking.sol	_____	1	69	34	15	27	15	_____
	packages/contracts/contracts/Interfaces/ICollSurplusPool.sol	_____	1	48	26	8	19	11	_____
 	packages/contracts/contracts/ZERO/SovStakersIssuance.sol	1	1	37	28	14	14	15	_____
	packages/contracts/contracts/ZERO/ZERO TokenStorage.sol	1	_____	54	54	28	11	26	_____

Type	File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
	packages/contracts/contracts/Interfaces/liquityBase.sol		1	14	10	5	3	5	
	Totals	44	17	7192	6146	3628	1725	2729	 

Legend: []

- **Lines:** total lines of the source unit
- **nLines:** normalized lines of the source unit (e.g. normalizes functions spanning multiple lines)
- **nSLOC:** normalized source lines of code (only source-code lines; no comments, no blank lines)
- **Comment Lines:** lines containing single or block comments
- **Complexity Score:** a custom complexity score derived from code statements that are known to introduce code complexity (branches, loops, calls, external interfaces, ...)

5. Scope of Work

The Sovryn Team provided us with the files that needs to be tested. The scope of the audit are the zero protocol contracts.

The team put forward the following assumptions regarding the security, usage of the contracts:

- The smart contract is coded according to the newest standards and in a secure way
- Each line of credit is required to be collateralized at a minimum of 110%
- Any owner of ZUSD can redeem their stablecoins for the underlying collateral at any time
- Lockup, staking and algorithmic issuance schedule of ZERO token is working as expected

The main goal of this audit was to verify these claims. The auditors can provide additional feedback on the code upon the client's request.

5.1 Manual and Automated Vulnerability Test

CRITICAL ISSUES

During the audit, Chainsulting's experts found **no Critical issues** in the code of the smart contract.

HIGH ISSUES

During the audit, Chainsulting's experts found **no High issues** in the code of the smart contract.

MEDIUM ISSUES

During the audit, Chainsulting's experts found **no Medium issues** in the code of the smart contract.

LOW ISSUES

5.1.1 Wrong import of OpenZeppelin library

Severity: LOW

Status: ACKNOWLEDGED

File(s) affected: All

Attack / Description	Code Snippet	Result/Recommendation
In the current implementation, OpenZeppelin files are added directly into the code. This violates OpenZeppelin's MIT license, which requires the license and copyright notice to	SafeMath, IERC20, Ownable	We highly recommend using npm (import "@openzeppelin/contracts/..") in order to guarantee that original OpenZeppelin contracts are used with no modifications. This also allows for any bug-fixes to be easily integrated into the codebase.


be included if its code is used. Moreover, updating code manually is error-prone.		
--	--	--

INFORMATIONAL ISSUES


During the audit, Chainsulting's experts found **no Informational issues** in the code of the smart contract.

5.2 Verify claims


5.2.1 The smart contract is coded according to the newest standards and in a secure way

Status: tested and verified 


5.2.2 Each line of credit is required to be collateralized at a minimum of 110%

Status: tested and verified 

5.2.3 Any owner of ZUSD can redeem their stablecoins for the underlying collateral at any time

Status: tested and verified 

5.2.4 Lockup, staking and algorithmic issuance schedule of ZERO token is working as expected

Status: tested and verified 

6. Executive Summary

Two (2) independent Chainsulting experts performed an unbiased and isolated audit of the smart contract codebase. The final debriefs took place on the October 13, 2021.

The main goal of the audit was to verify the claims regarding the security of the smart contract. During the audit, no critical issues were found, after the manual and automated security testing and the claim have been successfully verified.

Considering the complexity of zero protocol, the code quality that the Sovryn Team has presented, was exceptional. The inline comments have been helpful to understand the context and the unit tests covered already lots of possible test cases.

7. Deployed Smart Contract

PENDING

