HACKEN

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Customer: DeXe Network
Date: Jun 22, 2023

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Document

Name	Smart Contract Code Review and Security Analysis Report for DeXe Network
Type	DEX; Lending Platform; Voting; DAO
Platform	EVM
Language	Solidity
Methodology	Link
Website	https://dexe.network/
Changelog	08.03.2023 - Initial Review 12.04.2023 - Second Review 22.05.2023 - Third Review 22.06.2023 - Fourth Review

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Table of contents

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Introduction

Hacken OÜ (Consultant) was contracted by DeXe (Customer) to conduct a Smart Contract Code Review and Security Analysis. This report presents the findings of the security assessment of the Customer's smart contracts.

Scope

The scope of the project is review and security analysis of smart contracts in the repository:

Initial review scope

Repository	https://github.com/dexe-network/investment-contracts/tree/audit
Commit	f29a0e26ae59ffdf34f62e69ae1c32c606d10fd3
Whitepaper	Link
Functional Requirements	Not Provided
Technical Requirements	Not Provided
Contracts	File: ./contracts/gov/user-keeper/GovUserKeeper.sol SHA3: 4c18529e8623534c605cff1610dc6a0d1203fabb7e5fcbe57f17fceda0277d27 File: ./contracts/trader/TraderPoolRiskyProposal.sol SHA3: 13e541d548d958fe22818e4218e3144ad626ffda05a0433a22149781185d314c File: ./contracts/gov/GovPool.sol SHA3: 2c2258b6fc3723ae800737061177136d599c8d8257b6e7a2625cf76a4cd7e363 File: ./contracts/factory/PoolFactory.sol SHA3: a518d35ff6906e9f68e53c1692d56e0ea09fc32202cba4e6533794f6f7ab96cc File: ./contracts/gov/proposals/TokenSaleProposal.sol SHA3: 0d6d49b10c8d21f37aaa116662e27fd11c2607ddda07f3d21a87cf47c3485fc5 File: ./contracts/core/PriceFeed.sol SHA3: 07a1bac0dd37ed8ee3d1b2a13fcb64b8ef807350183bf35ba0fd37915e842f74 File: ./contracts/libs/trader-pool/TraderPoolView.sol SHA3: 50684b48437c911c5e86807663471bc130d17eb1d6f1979f793695179e077d22 File: ./contracts/trader/TraderPool.sol SHA3: f9f2e91d91d3473ebc94a1460b69288d0bc5a8867d7d75963bee428b2894f6ff File: ./contracts/trader/TraderPoolInvestProposal.sol SHA3: 7c0797ddce53469659eb4bfbcd819ac1530757c0fb84bc68b61e2e2b6de353a9 File: ./contracts/gov/validators/GovValidators.sol SHA3: cb8c8a99fcf8168d62f2ca6c1bb85b7250e9afb91d45326797d21b545e2b7ee0 File: ./contracts/libs/trader-pool-proposal/TraderPoolRiskyProposalView.sol SHA3: eb0af6263caaefba553002dd197c500c5077c6b88d580565b0c260ab9f2fec58

File: ./contracts/libs/gov-pool/GovPoolCreate.sol
SHA3: 3cd4f6eedf71ba34faf7014c7bdc140417383aa8b7fba19af3a28b2a71e5294e

File: ./contracts/trader/TraderPoolProposal.sol
SHA3: 734a22b8d70800fac7b5df9692334b1ee3911974433abd2767600a91daa3cd8b

File: ./contracts/core/CoreProperties.sol
SHA3: dc8e38b5fa764595d0bdbfa982f5110ac58c7b26e048ed405b2bf1a98b072322

File: ./contracts/libs/gov-user-keeper/GovUserKeeperView.sol
SHA3: d757f9f412995837294a6ca49d066d8765cd133d89b369d72fa06bea40dc145c

File: ./contracts/libs/trader-pool/TraderPoolCommission.sol
SHA3: 7b89bdfabf5f050f8b7b56ab4a626123388ba45bb73864c3f20be011ce51b9d8

File: ./contracts/gov/ERC721/ERC721Power.sol
SHA3: 83d98dd8965af73d07975625c166accc940d5e1cfc68b97635f81207bd3bf6d8

File: ./contracts/interfaces/gov/IGovPool.sol
SHA3: 2dd3c29a577aa6e9ed9d64b83a4e424c4b2e132672b6fc9cb32cd52bf7dff893

File: ./contracts/interfaces/gov/user-keeper/IGovUserKeeper.sol
SHA3: 48a13aa0718d98b9794518a27bda340be0444563bac867f66a9c8175661aaf4f

File: ./contracts/libs/gov-pool/GovPoolView.sol
SHA3: 5accf151ed9706a93b8e50fb5b7d26f4806b4d57c7ef4965011e9151988dd08e

File: ./contracts/libs/gov-pool/GovPoolStaking.sol
SHA3: b8e1c733f62d513743f4fbbb896f45f0905d8c7273e78460d52798f6acf1d690

File: ./contracts/interfaces/trader/ITraderPool.sol
SHA3: abb981c217b5effac122ea8db85c6f316fc5df5793a8604a978bec290ff2138e

File: ./contracts/libs/gov-pool/GovPoolVote.sol
SHA3: d16068da9f1b57e029f658fa82296f46a2e5d4f4d466d0d14c50e898b719b518

File: ./contracts/trader/InvestTraderPool.sol
SHA3: 19b50d98396aed5350b9c3c70bff02ff418df28c0f646b049e1a90b4cecc0768

File: ./contracts/trader/BasicTraderPool.sol
SHA3: c6ee6f683ef02e6ca5aab1f03b738bff3e19ad351f0359479c53b9f7afeccd5b

File: ./contracts/interfaces/trader/ITraderPoolRiskyProposal.sol
SHA3: 45cfdaacc98987e75049f869427ba381fd393e33a78ca056784b3bc4aca4fb7d

File: ./contracts/insurance/Insurance.sol
SHA3: 4e5f6fcd789c689e302e2fdf9f472fb4e5f699e611f4d0ad9e37f969542707ee

File: ./contracts/interfaces/core/ICoreProperties.sol
SHA3: 6edd8e9c2bde64a29392ef0332e7292f13bb3c7031006751db0dee8290229bfd

File: ./contracts/libs/trader-pool/TraderPoolDivest.sol
SHA3: d17c31adbedc13827442133183a7ca64cfacedcee19b0e65cfc71fa1181940e5

File: ./contracts/gov/settings/GovSettings.sol
SHA3: 17df5f43eef79fa87ee107c0256a0679fb070fc3b2e43fd27066572ee8fb4e59

File: ./contracts/libs/price-feed/UniswapV2PathFinder.sol
SHA3: 659f604ba4d8bd0ffa75965b64eac856650c688e82bd185fac4b8e8afdf3fb47

File: ./contracts/libs/trader-pool/TraderPoolInvest.sol
SHA3: f8fc1cd3ed9555738a831bd9c2ee89feda47019eac958f679eec6db8ac2d4364

	File: ./contracts/factory/PoolRegistry.sol SHA3: 94a40b62b6a42e610a2a5c56b84b874f51e3a2d2474e3dbade67b4885e172b82 File: ./contracts/libs/trader-pool-proposal/TraderPoolInvestProposalView.sol SHA3: c6bfc85f2fd3531fd06891cf3f96e0eaeb36a5c17519d1b58a05812d73c8f4d0 File: ./contracts/libs/trader-pool/TraderPoolExchange.sol SHA3: b0b16a5f392c18ea39b8839d2cd2f5b68c61062bc191d0a9181ff32f929f3bbc File: ./contracts/interfaces/core/IPriceFeed.sol SHA3: 31ed341e8c5b0b246115910ca34e72d4ddc0fed4f2b893d7daace88544734271 File: ./contracts/interfaces/gov/proposals/ITokenSaleProposal.sol SHA3: dfa3544e39e81468b7a1e9fc6504f9a08c47e9b8ba17ccc145c5122e2d3f9ba6 File: ./contracts/libs/gov-pool/GovPoolRewards.sol SHA3: 249431e98a61d9d296f2a8dcb1a3366a1c6dea2c1c589ef2ac017ef58ba12b87 File: ./contracts/interfaces/trader/ITraderPoolInvestProposal.sol SHA3: b7d0da23d9e0adca6bf19f219fe835a9458541266b326bfdd0b148e100d2683a File: ./contracts/gov/proposals/DistributionProposal.sol SHA3: d8575aad791aaead76a5bade09fc1f9d44d75102b95f85dd0158c49c0a2d348f File: ./contracts/gov/ERC721/ERC721Multiplier.sol SHA3: 432f60935aa2a746b23f99c5e32ea5e1cc1bb921b6fdb53caf1c584504439be4 File: ./contracts/interfaces/gov/validators/IGovValidators.sol SHA3: 54e58641fa16de2917a594dd8a73474dd43d50f5a55d3577fae904a868bf9a0d File: ./contracts/interfaces/factory/IPoolFactory.sol SHA3: a2fe28df6398836894cdc6b2f4faa0c87ca34fd071661f2d6d76db24286c55b5 File: ./contracts/libs/trader-pool/TraderPoolModify.sol SHA3: 56dad862ef096a1ba3677531750886c90c253d2f5af4388eb4e95b7ca12fd5fa File: ./contracts/libs/price-feed/PriceFeedLocal.sol SHA3: 5523af748ee8d1b6282213a165780f1356c27c4b467b5598a07b317cb34baaa8 File: ./contracts/libs/trader-pool/TraderPoolLeverage.sol SHA3: 4177bd110d85a43b6aebc719e7782dad5dcd7d496916672a9358437c2a2279e0 File: ./contracts/libs/gov-pool/GovPoolExecute.sol SHA3: 9834bf8ffa0448cf3923f66ca606f261be2b7727a516a6d471bfc8195a294735 File: ./contracts/gov/ERC20/ERC20Sale.sol SHA3: 6dada8f993335a243647bdf99dc25dd5220544bcabfa6621901ffc1fd2135e18 File: ./contracts/core/ContractsRegistry.sol SHA3: c650f3b1eaa611493cd8152707f5873def56e501163f0f345bb36ba07e462601 File: ./contracts/libs/gov-pool/GovPoolUnlock.sol SHA3: c5bfb3d67441f9cd8a372552c745a5f89559ca039b0c5df423aadce3f16d9ab2 File: ./contracts/libs/trader-pool/TraderPoolPrice.sol SHA3: aa63a767550e26db53298239f3f2df7ef7061f57acaf1c323337f7005029c0f3 File: ./contracts/user/UserRegistry.sol SHA3: de4e12194d2ec9d08f4362436a439752f30ccbd133727ec26a580194ed6b4298 File: ./contracts/libs/gov-pool/GovPoolOffchain.sol SHA3: 0ffe6530b31da6aa6e8ffbdd4d9b2e3076950f4f205e5a2df0b3b025e0c2563f

	File: ./contracts/interfaces/gov/settings/IGovSettings.sol SHA3: eb736a8c633a151fbfe2160706af5500d482deb90d5c36b64e8086994420a0b9 File: ./contracts/interfaces/factory/IPoolRegistry.sol SHA3: f9e1009febfdb55d2b0b5638b204ff12ace608d0635aea0ba221971f34406acc File: ./contracts/gov/validators/GovValidatorsToken.sol SHA3: 52e1acb8f2eaeae135b573dc9ce5ab4d791d6832daf7baf22065ec10df6590f1 File: ./contracts/libs/utils/TokenBalance.sol SHA3: 5d618e4e842a42325ee4151f163cb5cd97ddf3c58ea29d716236d8461b784a85 File: ./contracts/interfaces/trader/IBasicTraderPool.sol SHA3: 9c2b1a4412fad03f5a8d92475960bc6a0a05b86445cc9f7fa28534da76a8365c File: ./contracts/interfaces/insurance/IInsurance.sol SHA3: 72484bfb320f74c098433e953c8358f39f9afd1ac36cb163bad1cd80749dc7da File: ./contracts/libs/gov-pool/GovPoolCommission.sol SHA3: 9c97a54fdd17889e8449ce71519d44b28e73c316459079026bbbcb78453faaf9 File: ./contracts/libs/gov-user-keeper/GovUserKeeperLocal.sol SHA3: f2b4acb046fd0fd881683fd6555059e1544bd32731a6c13008b206fb188ec2d3 File: ./contracts/libs/data-structures/ShrinkableArray.sol SHA3: af91a726a7a6b755c49f670488b7a1c959e6b31357b05e1f59ababe77d1a19be File: ./contracts/interfaces/trader/IInvestTraderPool.sol SHA3: e19626a5fb21a651416eb69c00a97720c37ac18a09910a80c3ef1616cc620509 File: ./contracts/interfaces/gov/ERC721/IERC721Power.sol SHA3: 6eaca1243c20ec7728298d5e0990809352f0507a652a5ebcfabdbd622ae1927b File: ./contracts/libs/utils/AddressSetHelper.sol SHA3: 1075d40282f45bf5405709694dfbcdc76d073e3334fc2d69ff2fce713a9243c0 File: ./contracts/interfaces/trader/ITraderPoolProposal.sol SHA3: 9ed36ca1581a53a0ad32e32a1907cb0d941b01704a58582f38cf1906d8ba8312 File: ./contracts/libs/utils/DataHelper.sol SHA3: 78204bdb3cfb941360492a7dffd4c4a92879a89080214fa7279b5f9d45fffccd File: ./contracts/interfaces/gov/ERC20/IERC20Sale.sol SHA3: 535297f625b049fcaee902953a27bfcd8e23098e7c158ddea7cdfd93e99420cd File: ./contracts/interfaces/gov/ERC721/IERC721Multiplier.sol SHA3: 420ddb5bc2664a9fe4a2639e14da89b19be90cd9a5f2dc655d687b97496487c1 File: ./contracts/interfaces/core/IContractsRegistry.sol SHA3: 3f6ce4b25b64790f4da69355fc74035fdef5f2e1c9e5a4d383ed3afe347b621c File: ./contracts/interfaces/user/IUserRegistry.sol SHA3: 258ec5ad37eefeab690d1613f6616d668363df124c8634c730ec1d49941e651b File: ./contracts/interfaces/gov/proposals/IDistributionProposal.sol SHA3: eab572c444cc4b4acd3b90ed009f768eae8a3bf28b48390d9d879335da00688b File: ./contracts/libs/factory/GovTokenSaleDeployer.sol SHA3: 6c9adf2410c61afb735f014584ac4d7f9d7ea11ee8081e9748c7bfa057cec670 File: ./contracts/libs/math/MathHelper.sol SHA3: a1a0e0ab6a09d881b76020c9d707985a6dde6216387180f5ef1f328330ef71b3 File: ./contracts/core/Globals.sol

	SHA3: 24534da4cade47b84bd1cb443e5f07c8ef3e200389569020d8cc3ee662c65142 File: ./contracts/interfaces/gov/validators/IGovValidatorsToken.sol SHA3: eefa6f9c5b253f60354213156109f9858f743d9589ea1359224f2ab8121f5824 File: ./contracts/interfaces/trader/ITraderPoolMemberHook.sol SHA3: 1e4515d37c94a56d8b873b1567d645118798f75955d278a665d8585f7bc37b4a File: ./contracts/interfaces/core/ISBT721.sol SHA3: 5934aa0c2ad7779c0fa0b950a61b73418025dc258a289e3843211bc1d19995ac
Custom dependencies	https://www.npmjs.com/package/@dlsl/dev-modules/v/1.9.0: File: ./libs/arrays/Paginator.sol SHA3: 48d83bdd089aa9617e9a82248663cf09764ded57562552940e3e84928f661825 File: ./libs/arrays/ArrayHelper.sol SHA3: 1f1ce35efa7856505fbc42b0b16293009e1d0eec2fc0efa4428548b26fa9c9d7 File: ./pool-contracts-registry/pool-factory/AbstractPoolFactory.sol SHA3: 0b7e2428591f69b435de0c160dc95f41544410352495e87cf50ee444c3bd3762 File: ./libs/data-structures/StringSet.sol SHA3: 3e84c0d968e4e5945690ecd90417481a57ba92492c3309b6478416d436d103cc File: ./libs/decimals/DecimalsConverter.sol SHA3: 6b2d280ae4b550a3c6ee752508ad3c72bd9b2c3f2d9b4641f3ae3dccd7dfd2ba File: ./pool-contracts-registry/presets/OwnablePoolContractsRegistry.sol SHA3: dd035d49b28ef8d3806ff2ca5977bc4443fd4b2f34e4cebb2da3bcf5eba43db3 File: ./contracts-registry/AbstractDependant.sol SHA3: 4f0009cd3267becd8aff0a3e349b528c185d523f27de48a3e069f3cd3c0980ca

Second review scope

Repository	https://github.com/dexe-network/investment-contracts
Commit	24ce9a6f42648abd9460f679f7ed9dfe4a8665f5
Contracts	File: contracts/core/ContractsRegistry.sol SHA3: db542df2aca2967bc7ee93263cd3d111df33376577cb9c089badfaa141173e4e File: contracts/core/CoreProperties.sol SHA3: dc8e38b5fa764595d0bdbfa982f5110ac58c7b26e048ed405b2bf1a98b072322 File: contracts/core/Globals.sol SHA3: 24534da4cade47b84bd1cb443e5f07c8ef3e200389569020d8cc3ee662c65142 File: contracts/core/PriceFeed.sol SHA3: 07a1bac0dd37ed8ee3d1b2a13fcb64b8ef807350183bf35ba0fd37915e842f74 File: contracts/factory/PoolFactory.sol SHA3: 2ad5a7cce4fcd8f161f1a778c2b4dbff1f56e4e2c976d833227d75cef829133c File: contracts/factory/PoolRegistry.sol SHA3: 54c7cbbf77fbf1cd52e48c7c8b6fc6c5796a6a53f40c3e019e55ced845be29e5 File: contracts/gov/GovPool.sol

SHA3: 4008fcaa0d1e663a728c5aa64f90f1c77a8ba5cb0492e1b2ae194c23b341a77c

File: contracts/gov/ERC20/ERC20Sale.sol
SHA3: c84000a4382433555fc08d0d1dfae5558dc87ee1ff770ca4175a0fc2f22954a2

File: contracts/gov/ERC721/ERC721Multiplier.sol
SHA3: 5122010510a5895dfa86604b62e695d3f7b22f451e9eb677b4a29b314cb51438

File: contracts/gov/ERC721/ERC721Power.sol
SHA3: ff48489a427751fa8a094a902c5035fedb1e286f15fa3deb62d172eb21aebe3d

File: contracts/gov/proposals/DistributionProposal.sol
SHA3: 24f56842e144809a49b0108cb82c2b80419484424a8eed043ea4e7636d03c7d3

File: contracts/gov/proposals/TokenSaleProposal.sol
SHA3: 0fc3cd6aca808dc7a9fbb705b3724395909e2563343b2e85895e69350aa940ea

File: contracts/gov/settings/GovSettings.sol
SHA3: 17df5f43eef79fa87ee107c0256a0679fb070fc3b2e43fd27066572ee8fb4e59

File: contracts/gov/user-keeper/GovUserKeeper.sol
SHA3: 297546a6d1c55ee03690ccb5b43ec429da89bbf46d215748ff7aaa420a4b1f8d

File: contracts/gov/validators/GovValidators.sol
SHA3: 608933e41ec672c52408c7850ee98b11109e70b30efc540c89e924af83d25c1c

File: contracts/gov/validators/GovValidatorsToken.sol
SHA3: 94b536c4ffeca03a65182dd6b7e0f7e766ca533d24430e3084bfa56e88d65b3a

File: contracts/insurance/Insurance.sol
SHA3: 4e5f6fcd789c689e302e2fdf9f472fb4e5f699e611f4d0ad9e37f969542707ee

File: contracts/interfaces/core/IContractsRegistry.sol
SHA3: 3f6ce4b25b64790f4da69355fc74035fdef5f2e1c9e5a4d383ed3afe347b621c

File: contracts/interfaces/core/ICoreProperties.sol
SHA3: 6edd8e9c2bde64a29392ef0332e7292f13bb3c7031006751db0dee8290229bfd

File: contracts/interfaces/core/IPriceFeed.sol
SHA3: 31ed341e8c5b0b246115910ca34e72d4ddc0fed4f2b893d7daace88544734271

File: contracts/interfaces/factory/IPoolFactory.sol
SHA3: 0976fec9dca0e3cedb831c6a612e36af24b1eac86279fbb679eab494ee652008

File: contracts/interfaces/factory/IPoolRegistry.sol
SHA3: f9e1009febfdb55d2b0b5638b204ff12ace608d0635aea0ba221971f34406acc

File: contracts/interfaces/gov/IGovPool.sol
SHA3: 09088b9489d06fb629cf4444a5ebb77fe99482248b44cd00dd459234a80055a9

File: contracts/interfaces/gov/ERC20/IERC20Sale.sol
SHA3: c12887b5733cdd86dd0889ac60a313f405dff65eaf3dca3d04a4708788ef87c2

File: contracts/interfaces/gov/ERC721/IERC721Multiplier.sol
SHA3: 52d035625272a54ec9fc63542a60b89335b9b92803badc47e946adf3e289e7e1

File: contracts/interfaces/gov/ERC721/IERC721Power.sol
SHA3: 38c92048a1311871a82bbda8658cf984db36717c6df8640a7c33710908d7980d

File: contracts/interfaces/gov/proposals/IDistributionProposal.sol
SHA3: eab572c444cc4b4acd3b90ed009f768eae8a3bf28b48390d9d879335da00688b

File: contracts/interfaces/gov/proposals/ITokenSaleProposal.sol
SHA3: e63f02db1a0d2fda5a5d2008acaf1f1c6a7c92989cb07a73c81a899b25c1cfd2

File: contracts/interfaces/gov/settings/IGovSettings.sol
SHA3: eb736a8c633a151fbfe2160706af5500d482deb90d5c36b64e8086994420a0b9

File: contracts/interfaces/gov/user-keeper/IGovUserKeeper.sol
SHA3: c9d0d57d4a5ee6d74545dfeb13bf1f4d7f78614204037606fc8a69cc69077c48

File: contracts/interfaces/gov/validators/IGovValidators.sol
SHA3: a0d44a9a30358222d0f920b3a4e17f4190c579dfe686270a6631347aaea542b9

File: contracts/interfaces/gov/validators/IGovValidatorsToken.sol
SHA3: eefa6f9c5b253f60354213156109f9858f743d9589ea1359224f2ab8121f5824

File: contracts/interfaces/insurance/IInsurance.sol
SHA3: 72484bfb320f74c098433e953c8358f39f9afd1ac36cb163bad1cd80749dc7da

File: contracts/interfaces/trader/IBasicTraderPool.sol
SHA3: 9c2b1a4412fad03f5a8d92475960bc6a0a05b86445cc9f7fa28534da76a8365c

File: contracts/interfaces/trader/IInvestTraderPool.sol
SHA3: e19626a5fb21a651416eb69c00a97720c37ac18a09910a80c3ef1616cc620509

File: contracts/interfaces/trader/ITraderPool.sol
SHA3: 1f957cdd16d88a1b92b58c430dda6a54216b8275b2c198d59601655bfdd8bc94

File: contracts/interfaces/trader/ITraderPoolInvestProposal.sol
SHA3: b7d0da23d9e0adca6bf19f219fe835a9458541266b326bfdd0b148e100d2683a

File: contracts/interfaces/trader/ITraderPoolMemberHook.sol
SHA3: 1e4515d37c94a56d8b873b1567d645118798f75955d278a665d8585f7bc37b4a

File: contracts/interfaces/trader/ITraderPoolProposal.sol
SHA3: 9ed36ca1581a53a0ad32e32a1907cb0d941b01704a58582f38cf1906d8ba8312

File: contracts/interfaces/trader/ITraderPoolRiskyProposal.sol
SHA3: 45cfdaacc98987e75049f869427ba381fd393e33a78ca056784b3bc4aca4fb7d

File: contracts/interfaces/user/IUserRegistry.sol
SHA3: 258ec5ad37eefeab690d1613f6616d668363df124c8634c730ec1d49941e651b

File: contracts/libs/data-structures/ShrinkableArray.sol
SHA3: af91a726a7a6b755c49f670488b7a1c959e6b31357b05e1f59ababe77d1a19be

File: contracts/libs/factory/GovTokenSaleDeployer.sol
SHA3: d38ffd486b4e68583aac28838c49c5cbe163d53dae2704ae82befafa7371b743

File: contracts/libs/gov-pool/GovPoolCommission.sol
SHA3: 9c97a54fdd17889e8449ce71519d44b28e73c316459079026bbbcb78453faaf9

File: contracts/libs/gov-pool/GovPoolCreate.sol
SHA3: 5f353d2881933d1c8e6eba345fecc2b1d33da7dcca1abab70a7de4f2a8cef8d4

File: contracts/libs/gov-pool/GovPoolExecute.sol
SHA3: 8a09c66755a54c890da9ad05915a01b8dff6f825665942e57ac600403507a5a0

File: contracts/libs/gov-pool/GovPoolOffchain.sol
SHA3: 0ffe6530b31da6aa6e8ffbdd4d9b2e3076950f4f205e5a2df0b3b025e0c2563f

File: contracts/libs/gov-pool/GovPoolRewards.sol
SHA3: 00947e9d18e4c156ce9f8f7c7aec2e7c645bfe976b7c47d92509523f3366fb37

File: contracts/libs/gov-pool/GovPoolStaking.sol
SHA3: 70700059fad91d78b5bc0737db576dbb167fa9c4dd9e3b6341525dc1088bda0c

	File: contracts/libs/gov-pool/GovPoolUnlock.sol SHA3: c5bfb3d67441f9cd8a372552c745a5f89559ca039b0c5df423aadce3f16d9ab2 File: contracts/libs/gov-pool/GovPoolView.sol SHA3: 5accf151ed9706a93b8e50fb5b7d26f4806b4d57c7ef4965011e9151988dd08e File: contracts/libs/gov-pool/GovPoolVote.sol SHA3: 8c3e5ad715e72c691f1b750fe985fe4b9f7db490c8c8650b2e7767cf7e01a182 File: contracts/libs/gov-user-keeper/GovUserKeeperLocal.sol SHA3: f2b4acb046fd0fd881683fd6555059e1544bd32731a6c13008b206fb188ec2d3 File: contracts/libs/gov-user-keeper/GovUserKeeperView.sol SHA3: d757f9f412995837294a6ca49d066d8765cd133d89b369d72fa06bea40dc145c File: contracts/libs/math/MathHelper.sol SHA3: a1a0e0ab6a09d881b76020c9d707985a6dde6216387180f5ef1f328330ef71b3 File: contracts/libs/price-feed/PriceFeedLocal.sol SHA3: 5523af748ee8d1b6282213a165780f1356c27c4b467b5598a07b317cb34baaa8 File: contracts/libs/price-feed/UniswapV2PathFinder.sol SHA3: 659f604ba4d8bd0ffa75965b64eac856650c688e82bd185fac4b8e8afdf3fb47 File: contracts/libs/trader-pool/TraderPoolCommission.sol SHA3: 7b89bdfabf5f050f8b7b56ab4a626123388ba45bb73864c3f20be011ce51b9d8 File: contracts/libs/trader-pool/TraderPoolDivest.sol SHA3: 5a8b3a432cb70b91807f2e87dd370ffbf26672b9085c172b8bbaa29e4c736624 File: contracts/libs/trader-pool/TraderPoolExchange.sol SHA3: 5e7b89ae8fb477cd2e646d83fa5246af95d7c9e5bd7ca7cf9a10d7e405ca975f File: contracts/libs/trader-pool/TraderPoolInvest.sol SHA3: 665eb443cee0ddcbf17ad781a296f2dde81361c7d7565c5cf9685e3064aa71ee File: contracts/libs/trader-pool/TraderPoolLeverage.sol SHA3: 4177bd110d85a43b6aebc719e7782dad5dcd7d496916672a9358437c2a2279e0 File: contracts/libs/trader-pool/TraderPoolModify.sol SHA3: 0774660e8a2801632b9e0a95aea532d82b05c28bafb1f1b9615593072e429079 File: contracts/libs/trader-pool/TraderPoolPrice.sol SHA3: aa63a767550e26db53298239f3f2df7ef7061f57acaf1c323337f7005029c0f3 File: contracts/libs/trader-pool/TraderPoolView.sol SHA3: adc624a80827a9017040929bd3a2edf6334e1ec9ac77c2a037c5a5cbb1f97d3e File: contracts/libs/trader-pool-proposal/TraderPoolInvestProposalView.sol SHA3: c6bfc85f2fd3531fd06891cf3f96e0eaeb36a5c17519d1b58a05812d73c8f4d0 File: contracts/libs/trader-pool-proposal/TraderPoolRiskyProposalView.sol SHA3: eb0af6263caaefba553002dd197c500c5077c6b88d580565b0c260ab9f2fec58 File: contracts/libs/utils/AddressSetHelper.sol SHA3: 1075d40282f45bf5405709694dfbcdc76d073e3334fc2d69ff2fce713a9243c0 File: contracts/libs/utils/DataHelper.sol SHA3: 78204bdb3cfb941360492a7dffd4c4a92879a89080214fa7279b5f9d45fffccd File: contracts/libs/utils/TokenBalance.sol SHA3: 5d618e4e842a42325ee4151f163cb5cd97ddf3c58ea29d716236d8461b784a85

	File: contracts/trader/BasicTraderPool.sol SHA3: 1ac45b856a2326afe17087f145745c297c548ab0698fb4f9b095fb6eeedf9797 File: contracts/trader/InvestTraderPool.sol SHA3: 4309fc11902d9f0bdff97a2f4890ec8bdd5f77773502214ac1cccd8de316fb98 File: contracts/trader/TraderPool.sol SHA3: c86a797db704507f3d78c9fe698a375a392d7608ff88dcae67a2d4b6713d23e2 File: contracts/trader/TraderPoolInvestProposal.sol SHA3: 7c0797ddce53469659eb4bfbcd819ac1530757c0fb84bc68b61e2e2b6de353a9 File: contracts/trader/TraderPoolProposal.sol SHA3: ab30267a8ac8d078d8e40a56fcbf2d0421d3ecf87593ebafb43699f4d8517e8b File: contracts/trader/TraderPoolRiskyProposal.sol SHA3: 79a26b4bad51f46fb4d73da939250ca06a6f363a87611865b826800de32b0af3 File: contracts/user/UserRegistry.sol SHA3: a5b4ff5e7199214a1be67ff8201f83792b2638013594257087ec276669937503

Third review scope

Repository	https://github.com/dexe-network/investment-contracts
Commit	3c04c3e063feb958b9b7ac7a112b705a58b0832d
Contracts	File: contracts/core/ContractsRegistry.sol SHA3: c650f3b1eaa611493cd8152707f5873def56e501163f0f345bb36ba07e462601 File: contracts/core/CoreProperties.sol SHA3: dc8e38b5fa764595d0bdbfa982f5110ac58c7b26e048ed405b2bf1a98b072322 File: contracts/core/Globals.sol SHA3: 24534da4cade47b84bd1cb443e5f07c8ef3e200389569020d8cc3ee662c65142 File: contracts/core/PriceFeed.sol SHA3: 07a1bac0dd37ed8ee3d1b2a13fcb64b8ef807350183bf35ba0fd37915e842f74 File: contracts/factory/PoolFactory.sol SHA3: a518d35ff6906e9f68e53c1692d56e0ea09fc32202cba4e6533794f6f7ab96cc File: contracts/factory/PoolRegistry.sol SHA3: 94a40b62b6a42e610a2a5c56b84b874f51e3a2d2474e3dbade67b4885e172b82 File: contracts/gov/GovPool.sol SHA3: 2c2258b6fc3723ae800737061177136d599c8d8257b6e7a2625cf76a4cd7e363 File: contracts/gov/ERC20/ERC20Sale.sol SHA3: 6dada8f993335a243647bdf99dc25dd5220544bcabfa6621901ffc1fd2135e18 File: contracts/gov/ERC721/ERC721Multiplier.sol SHA3: 432f60935aa2a746b23f99c5e32ea5e1cc1bb921b6fdb53caf1c584504439be4 File: contracts/gov/ERC721/ERC721Power.sol SHA3: 83d98dd8965af73d07975625c166accc940d5e1cfc68b97635f81207bd3bf6d8 File: contracts/gov/proposals/DistributionProposal.sol SHA3: d8575aad791aaead76a5bade09fc1f9d44d75102b95f85dd0158c49c0a2d348f File: contracts/gov/proposals/TokenSaleProposal.sol

SHA3: 0d6d49b10c8d21f37aaa116662e27fd11c2607ddda07f3d21a87cf47c3485fc5

File: contracts/gov/settings/GovSettings.sol
SHA3: 17df5f43eef79fa87ee107c0256a0679fb070fc3b2e43fd27066572ee8fb4e59

File: contracts/gov/user-keeper/GovUserKeeper.sol
SHA3: 4c18529e8623534c605cff1610dc6a0d1203fabb7e5fcbe57f17fceda0277d27

File: contracts/gov/validators/GovValidators.sol
SHA3: cb8c8a99fcf8168d62f2ca6c1bb85b7250e9afb91d45326797d21b545e2b7ee0

File: contracts/gov/validators/GovValidatorsToken.sol
SHA3: 52e1acb8f2eaeae135b573dc9ce5ab4d791d6832daf7baf22065ec10df6590f1

File: contracts/insurance/Insurance.sol
SHA3: 4e5f6fcd789c689e302e2fdf9f472fb4e5f699e611f4d0ad9e37f969542707ee

File: contracts/interfaces/core/IContractsRegistry.sol
SHA3: 3f6ce4b25b64790f4da69355fc74035fdef5f2e1c9e5a4d383ed3afe347b621c

File: contracts/interfaces/core/ICoreProperties.sol
SHA3: 6edd8e9c2bde64a29392ef0332e7292f13bb3c7031006751db0dee8290229bfd

File: contracts/interfaces/core/IPriceFeed.sol
SHA3: 31ed341e8c5b0b246115910ca34e72d4ddc0fed4f2b893d7daace88544734271

File: contracts/interfaces/factory/IPoolFactory.sol
SHA3: a2fe28df6398836894cdc6b2f4faa0c87ca34fd071661f2d6d76db24286c55b5

File: contracts/interfaces/factory/IPoolRegistry.sol
SHA3: f9e1009febfdb55d2b0b5638b204ff12ace608d0635aea0ba221971f34406acc

File: contracts/interfaces/gov/IGovPool.sol
SHA3: 2dd3c29a577aa6e9ed9d64b83a4e424c4b2e132672b6fc9cb32cd52bf7dff893

File: contracts/interfaces/gov/ERC20/IERC20Sale.sol
SHA3: 535297f625b049fcaee902953a27bfcd8e23098e7c158ddea7cdfd93e99420cd

File: contracts/interfaces/gov/ERC721/IERC721Multiplier.sol
SHA3: 420ddb5bc2664a9fe4a2639e14da89b19be90cd9a5f2dc655d687b97496487c1

File: contracts/interfaces/gov/ERC721/IERC721Power.sol
SHA3: 6eaca1243c20ec7728298d5e0990809352f0507a652a5ebcfabdbd622ae1927b

File: contracts/interfaces/gov/proposals/IDistributionProposal.sol
SHA3: eab572c444cc4b4acd3b90ed009f768eae8a3bf28b48390d9d879335da00688b

File: contracts/interfaces/gov/proposals/ITokenSaleProposal.sol
SHA3: dfa3544e39e81468b7a1e9fc6504f9a08c47e9b8ba17ccc145c5122e2d3f9ba6

File: contracts/interfaces/gov/settings/IGovSettings.sol
SHA3: eb736a8c633a151fbfe2160706af5500d482deb90d5c36b64e8086994420a0b9

File: contracts/interfaces/gov/user-keeper/IGovUserKeeper.sol
SHA3: 48a13aa0718d98b9794518a27bda340be0444563bac867f66a9c8175661aaf4f

File: contracts/interfaces/gov/validators/IGovValidators.sol
SHA3: 54e58641fa16de2917a594dd8a73474dd43d50f5a55d3577fae904a868bf9a0d

File: contracts/interfaces/gov/validators/IGovValidatorsToken.sol
SHA3: eefa6f9c5b253f60354213156109f9858f743d9589ea1359224f2ab8121f5824

File: contracts/interfaces/insurance/IInsurance.sol
SHA3: 72484bfb320f74c098433e953c8358f39f9afd1ac36cb163bad1cd80749dc7da

File: contracts/interfaces/trader/IBasicTraderPool.sol
SHA3: 9c2b1a4412fad03f5a8d92475960bc6a0a05b86445cc9f7fa28534da76a8365c

File: contracts/interfaces/trader/IInvestTraderPool.sol
SHA3: e19626a5fb21a651416eb69c00a97720c37ac18a09910a80c3ef1616cc620509

File: contracts/interfaces/trader/ITraderPool.sol
SHA3: abb981c217b5effac122ea8db85c6f316fc5df5793a8604a978bec290ff2138e

File: contracts/interfaces/trader/ITraderPoolInvestProposal.sol
SHA3: b7d0da23d9e0adca6bf19f219fe835a9458541266b326bfdd0b148e100d2683a

File: contracts/interfaces/trader/ITraderPoolMemberHook.sol
SHA3: 1e4515d37c94a56d8b873b1567d645118798f75955d278a665d8585f7bc37b4a

File: contracts/interfaces/trader/ITraderPoolProposal.sol
SHA3: 9ed36ca1581a53a0ad32e32a1907cb0d941b01704a58582f38cf1906d8ba8312

File: contracts/interfaces/trader/ITraderPoolRiskyProposal.sol
SHA3: 45cfdaacc98987e75049f869427ba381fd393e33a78ca056784b3bc4aca4fb7d

File: contracts/interfaces/user/IUserRegistry.sol
SHA3: 258ec5ad37eefeab690d1613f6616d668363df124c8634c730ec1d49941e651b

File: contracts/libs/data-structures/ShrinkableArray.sol
SHA3: af91a726a7a6b755c49f670488b7a1c959e6b31357b05e1f59ababe77d1a19be

File: contracts/libs/factory/GovTokenSaleDeployer.sol
SHA3: 6c9adf2410c61afb735f014584ac4d7f9d7ea11ee8081e9748c7bfa057cec670

File: contracts/libs/gov-pool/GovPoolCommission.sol
SHA3: 9c97a54fdd17889e8449ce71519d44b28e73c316459079026bbbcb78453faaf9

File: contracts/libs/gov-pool/GovPoolCreate.sol
SHA3: 3cd4f6eedf71ba34faf7014c7bdc140417383aa8b7fba19af3a28b2a71e5294e

File: contracts/libs/gov-pool/GovPoolExecute.sol
SHA3: 9834bf8ffa0448cf3923f66ca606f261be2b7727a516a6d471bfc8195a294735

File: contracts/libs/gov-pool/GovPoolOffchain.sol
SHA3: 0ffe6530b31da6aa6e8ffbdd4d9b2e3076950f4f205e5a2df0b3b025e0c2563f

File: contracts/libs/gov-pool/GovPoolRewards.sol
SHA3: 249431e98a61d9d296f2a8dcb1a3366a1c6dea2c1c589ef2ac017ef58ba12b87

File: contracts/libs/gov-pool/GovPoolStaking.sol
SHA3: b8e1c733f62d513743f4fbbb896f45f0905d8c7273e78460d52798f6acf1d690

File: contracts/libs/gov-pool/GovPoolUnlock.sol
SHA3: c5bfb3d67441f9cd8a372552c745a5f89559ca039b0c5df423aadce3f16d9ab2

File: contracts/libs/gov-pool/GovPoolView.sol
SHA3: 5accf151ed9706a93b8e50fb5b7d26f4806b4d57c7ef4965011e9151988dd08e

File: contracts/libs/gov-pool/GovPoolVote.sol
SHA3: d16068da9f1b57e029f658fa82296f46a2e5d4f4d466d0d14c50e898b719b518

File: contracts/libs/gov-user-keeper/GovUserKeeperLocal.sol
SHA3: f2b4acb046fd0fd881683fd6555059e1544bd32731a6c13008b206fb188ec2d3

File: contracts/libs/gov-user-keeper/GovUserKeeperView.sol
SHA3: d757f9f412995837294a6ca49d066d8765cd133d89b369d72fa06bea40dc145c

	File: contracts/libs/math/MathHelper.sol SHA3: a1a0e0ab6a09d881b76020c9d707985a6dde6216387180f5ef1f328330ef71b3 File: contracts/libs/price-feed/PriceFeedLocal.sol SHA3: 5523af748ee8d1b6282213a165780f1356c27c4b467b5598a07b317cb34baaa8 File: contracts/libs/price-feed/UniswapV2PathFinder.sol SHA3: 659f604ba4d8bd0ffa75965b64eac856650c688e82bd185fac4b8e8afdf3fb47 File: contracts/libs/trader-pool/TraderPoolCommission.sol SHA3: 7b89bdfabf5f050f8b7b56ab4a626123388ba45bb73864c3f20be011ce51b9d8 File: contracts/libs/trader-pool/TraderPoolDivest.sol SHA3: d17c31adbedc13827442133183a7ca64cfacedcee19b0e65cfc71fa1181940e5 File: contracts/libs/trader-pool/TraderPoolExchange.sol SHA3: b0b16a5f392c18ea39b8839d2cd2f5b68c61062bc191d0a9181ff32f929f3bbc File: contracts/libs/trader-pool/TraderPoolInvest.sol SHA3: f8fc1cd3ed9555738a831bd9c2ee89feda47019eac958f679eec6db8ac2d4364 File: contracts/libs/trader-pool/TraderPoolLeverage.sol SHA3: 4177bd110d85a43b6aebc719e7782dad5dcd7d496916672a9358437c2a2279e0 File: contracts/libs/trader-pool/TraderPoolModify.sol SHA3: 56dad862ef096a1ba3677531750886c90c253d2f5af4388eb4e95b7ca12fd5fa File: contracts/libs/trader-pool/TraderPoolPrice.sol SHA3: aa63a767550e26db53298239f3f2df7ef7061f57acaf1c323337f7005029c0f3 File: contracts/libs/trader-pool/TraderPoolView.sol SHA3: 50684b48437c911c5e86807663471bc130d17eb1d6f1979f793695179e077d22 File: contracts/libs/trader-pool-proposal/TraderPoolInvestProposalView.sol SHA3: c6bfc85f2fd3531fd06891cf3f96e0eaeb36a5c17519d1b58a05812d73c8f4d0 File: contracts/libs/trader-pool-proposal/TraderPoolRiskyProposalView.sol SHA3: eb0af6263caaefba553002dd197c500c5077c6b88d580565b0c260ab9f2fec58 File: contracts/libs/utils/AddressSetHelper.sol SHA3: 1075d40282f45bf5405709694dfbcdc76d073e3334fc2d69ff2fce713a9243c0 File: contracts/libs/utils/DataHelper.sol SHA3: 78204bdb3cfb941360492a7dffd4c4a92879a89080214fa7279b5f9d45fffccd File: contracts/libs/utils/TokenBalance.sol SHA3: 5d618e4e842a42325ee4151f163cb5cd97ddf3c58ea29d716236d8461b784a85 File: contracts/trader/BasicTraderPool.sol SHA3: c6ee6f683ef02e6ca5aab1f03b738bff3e19ad351f0359479c53b9f7afeccd5b File: contracts/trader/InvestTraderPool.sol SHA3: 19b50d98396aed5350b9c3c70bff02ff418df28c0f646b049e1a90b4cecc0768 File: contracts/trader/TraderPool.sol SHA3: f9f2e91d91d3473ebc94a1460b69288d0bc5a8867d7d75963bee428b2894f6ff File: contracts/trader/TraderPoolInvestProposal.sol SHA3: 7c0797ddce53469659eb4bfbcd819ac1530757c0fb84bc68b61e2e2b6de353a9 File: contracts/trader/TraderPoolProposal.sol SHA3: 734a22b8d70800fac7b5df9692334b1ee3911974433abd2767600a91daa3cd8b

	File: contracts/trader/TraderPoolRiskyProposal.sol SHA3: 13e541d548d958fe22818e4218e3144ad626ffda05a0433a22149781185d314c File: contracts/user/UserRegistry.sol SHA3: de4e12194d2ec9d08f4362436a439752f30ccbd133727ec26a580194ed6b4298

Fourth review scope

Repository	https://github.com/dexe-network/investment-contracts
Commit	bd1044eba1007aa2aceed28b4e9e6a070aa7ef5a
Contracts	File: contracts/core/ContractsRegistry.sol SHA3: db542df2aca2967bc7ee93263cd3d111df33376577cb9c089badfaa141173e4e File: contracts/core/CoreProperties.sol SHA3: dc8e38b5fa764595d0bdbfa982f5110ac58c7b26e048ed405b2bf1a98b072322 File: contracts/core/Globals.sol SHA3: 24534da4cade47b84bd1cb443e5f07c8ef3e200389569020d8cc3ee662c65142 File: contracts/core/PriceFeed.sol SHA3: 07a1bac0dd37ed8ee3d1b2a13fcb64b8ef807350183bf35ba0fd37915e842f74 File: contracts/factory/PoolFactory.sol SHA3: 2ad5a7cce4fcd8f161f1a778c2b4dbff1f56e4e2c976d833227d75cef829133c File: contracts/factory/PoolRegistry.sol SHA3: 54c7cbbf77fbf1cd52e48c7c8b6fc6c5796a6a53f40c3e019e55ced845be29e5 File: contracts/gov/GovPool.sol SHA3: 4008fcaa0d1e663a728c5aa64f90f1c77a8ba5cb0492e1b2ae194c23b341a77c File: contracts/gov/ERC20/ERC20Sale.sol SHA3: c84000a4382433555fc08d0d1dfae5558dc87ee1ff770ca4175a0fc2f22954a2 File: contracts/gov/ERC721/ERC721Multiplier.sol SHA3: 5122010510a5895dfa86604b62e695d3f7b22f451e9eb677b4a29b314cb51438 File: contracts/gov/ERC721/ERC721Power.sol SHA3: ff48489a427751fa8a094a902c5035fedb1e286f15fa3deb62d172eb21aebe3d File: contracts/gov/proposals/DistributionProposal.sol SHA3: 24f56842e144809a49b0108cb82c2b80419484424a8eed043ea4e7636d03c7d3 File: contracts/gov/proposals/TokenSaleProposal.sol SHA3: 6408df88e86898ef1cbe74f0786e07f6daf408033a88b12e871baee8985d1a4e File: contracts/gov/settings/GovSettings.sol SHA3: 17df5f43eef79fa87ee107c0256a0679fb070fc3b2e43fd27066572ee8fb4e59 File: contracts/gov/user-keeper/GovUserKeeper.sol SHA3: 172149bab6bc7fbf02a060c560734509f6587b1dc6c97fcc03db39435905702e File: contracts/gov/validators/GovValidators.sol SHA3: 608933e41ec672c52408c7850ee98b11109e70b30efc540c89e924af83d25c1c File: contracts/gov/validators/GovValidatorsToken.sol SHA3: 94b536c4ffeca03a65182dd6b7e0f7e766ca533d24430e3084bfa56e88d65b3a File: contracts/insurance/Insurance.sol

SHA3: 4e5f6fcd789c689e302e2fdf9f472fb4e5f699e611f4d0ad9e37f969542707ee

File: contracts/interfaces/core/IContractsRegistry.sol
SHA3: 3f6ce4b25b64790f4da69355fc74035fdef5f2e1c9e5a4d383ed3afe347b621c

File: contracts/interfaces/core/ICoreProperties.sol
SHA3: 6edd8e9c2bde64a29392ef0332e7292f13bb3c7031006751db0dee8290229bfd

File: contracts/interfaces/core/IPriceFeed.sol
SHA3: 31ed341e8c5b0b246115910ca34e72d4ddc0fed4f2b893d7daace88544734271

File: contracts/interfaces/factory/IPoolFactory.sol
SHA3: 0976fec9dca0e3cedb831c6a612e36af24b1eac86279fbb679eab494ee652008

File: contracts/interfaces/factory/IPoolRegistry.sol
SHA3: f9e1009febfdb55d2b0b5638b204ff12ace608d0635aea0ba221971f34406acc

File: contracts/interfaces/gov/IGovPool.sol
SHA3: 09088b9489d06fb629cf4444a5ebb77fe99482248b44cd00dd459234a80055a9

File: contracts/interfaces/gov/ERC20/IERC20Sale.sol
SHA3: c12887b5733cdd86dd0889ac60a313f405dff65eaf3dca3d04a4708788ef87c2

File: contracts/interfaces/gov/ERC721/IERC721Multiplier.sol
SHA3: 52d035625272a54ec9fc63542a60b89335b9b92803badc47e946adf3e289e7e1

File: contracts/interfaces/gov/ERC721/IERC721Power.sol
SHA3: 38c92048a1311871a82bbda8658cf984db36717c6df8640a7c33710908d7980d

File: contracts/interfaces/gov/proposals/IDistributionProposal.sol
SHA3: eab572c444cc4b4acd3b90ed009f768eae8a3bf28b48390d9d879335da00688b

File: contracts/interfaces/gov/proposals/ITokenSaleProposal.sol
SHA3: b610e2615e689ab9c6b7cfa74695de2b161cc4283ef5fd0e4288019ea4ab58db

File: contracts/interfaces/gov/settings/IGovSettings.sol
SHA3: eb736a8c633a151fbfe2160706af5500d482deb90d5c36b64e8086994420a0b9

File: contracts/interfaces/gov/user-keeper/IGovUserKeeper.sol
SHA3: 2cd13c7f98275e3073d18f74be9cda5299429cd7b1ef0921f174cc3674f4e225

File: contracts/interfaces/gov/validators/IGovValidators.sol
SHA3: a0d44a9a30358222d0f920b3a4e17f4190c579dfe686270a6631347aaea542b9

File: contracts/interfaces/gov/validators/IGovValidatorsToken.sol
SHA3: eefa6f9c5b253f60354213156109f9858f743d9589ea1359224f2ab8121f5824

File: contracts/interfaces/insurance/IInsurance.sol
SHA3: 72484bfb320f74c098433e953c8358f39f9afd1ac36cb163bad1cd80749dc7da

File: contracts/interfaces/trader/IBasicTraderPool.sol
SHA3: 9c2b1a4412fad03f5a8d92475960bc6a0a05b86445cc9f7fa28534da76a8365c

File: contracts/interfaces/trader/IInvestTraderPool.sol
SHA3: e19626a5fb21a651416eb69c00a97720c37ac18a09910a80c3ef1616cc620509

File: contracts/interfaces/trader/ITraderPool.sol
SHA3: 1f957cdd16d88a1b92b58c430dda6a54216b8275b2c198d59601655bfdd8bc94

File: contracts/interfaces/trader/ITraderPoolInvestProposal.sol
SHA3: b7d0da23d9e0adca6bf19f219fe835a9458541266b326bfdd0b148e100d2683a

File: contracts/interfaces/trader/ITraderPoolMemberHook.sol
SHA3: 1e4515d37c94a56d8b873b1567d645118798f75955d278a665d8585f7bc37b4a

File: contracts/interfaces/trader/ITraderPoolProposal.sol
SHA3: 9ed36ca1581a53a0ad32e32a1907cb0d941b01704a58582f38cf1906d8ba8312

File: contracts/interfaces/trader/ITraderPoolRiskyProposal.sol
SHA3: 45cfdaacc98987e75049f869427ba381fd393e33a78ca056784b3bc4aca4fb7d

File: contracts/interfaces/user/IUserRegistry.sol
SHA3: 258ec5ad37eefeab690d1613f6616d668363df124c8634c730ec1d49941e651b

File: contracts/libs/data-structures/ShrinkableArray.sol
SHA3: af91a726a7a6b755c49f670488b7a1c959e6b31357b05e1f59ababe77d1a19be

File: contracts/libs/factory/GovTokenSaleDeployer.sol
SHA3: d38ffd486b4e68583aac28838c49c5cbe163d53dae2704ae82befafa7371b743

File: contracts/libs/gov-pool/GovPoolCommission.sol
SHA3: 9c97a54fdd17889e8449ce71519d44b28e73c316459079026bbbcb78453faaf9

File: contracts/libs/gov-pool/GovPoolCreate.sol
SHA3: 0c1000ef0b283bb5e90217808df4e78d21936e42b7f8d0458b8c67bb25895394

File: contracts/libs/gov-pool/GovPoolExecute.sol
SHA3: 8a09c66755a54c890da9ad05915a01b8dff6f825665942e57ac600403507a5a0

File: contracts/libs/gov-pool/GovPoolOffchain.sol
SHA3: b69c0f9f61ab9335151fad76338fca7c6dc192d9a547c53a6f8660b67230419e

File: contracts/libs/gov-pool/GovPoolRewards.sol
SHA3: 00947e9d18e4c156ce9f8f7c7aec2e7c645bfe976b7c47d92509523f3366fb37

File: contracts/libs/gov-pool/GovPoolStaking.sol
SHA3: 70700059fad91d78b5bc0737db576dbb167fa9c4dd9e3b6341525dc1088bda0c

File: contracts/libs/gov-pool/GovPoolUnlock.sol
SHA3: c5bfb3d67441f9cd8a372552c745a5f89559ca039b0c5df423aadce3f16d9ab2

File: contracts/libs/gov-pool/GovPoolView.sol
SHA3: 5accf151ed9706a93b8e50fb5b7d26f4806b4d57c7ef4965011e9151988dd08e

File: contracts/libs/gov-pool/GovPoolVote.sol
SHA3: 13dde5380f8e480e5d5dd22ca46659fe2459ec0c4c2fcd2c68785667c8d34168

File: contracts/libs/gov-user-keeper/GovUserKeeperLocal.sol
SHA3: f2b4acb046fd0fd881683fd6555059e1544bd32731a6c13008b206fb188ec2d3

File: contracts/libs/gov-user-keeper/GovUserKeeperView.sol
SHA3: d757f9f412995837294a6ca49d066d8765cd133d89b369d72fa06bea40dc145c

File: contracts/libs/math/MathHelper.sol
SHA3: a1a0e0ab6a09d881b76020c9d707985a6dde6216387180f5ef1f328330ef71b3

File: contracts/libs/price-feed/PriceFeedLocal.sol
SHA3: 5523af748ee8d1b6282213a165780f1356c27c4b467b5598a07b317cb34baaa8

File: contracts/libs/price-feed/UniswapV2PathFinder.sol
SHA3: 659f604ba4d8bd0ffa75965b64eac856650c688e82bd185fac4b8e8afdf3fb47

File: contracts/libs/trader-pool/TraderPoolCommission.sol
SHA3: 7b89bdfabf5f050f8b7b56ab4a626123388ba45bb73864c3f20be011ce51b9d8

File: contracts/libs/trader-pool/TraderPoolDivest.sol
SHA3: 5a8b3a432cb70b91807f2e87dd370ffbf26672b9085c172b8bbaa29e4c736624

	File: contracts/libs/trader-pool/TraderPoolExchange.sol SHA3: 5e7b89ae8fb477cd2e646d83fa5246af95d7c9e5bd7ca7cf9a10d7e405ca975f File: contracts/libs/trader-pool/TraderPoolInvest.sol SHA3: 665eb443cee0ddcbf17ad781a296f2dde81361c7d7565c5cf9685e3064aa71ee File: contracts/libs/trader-pool/TraderPoolLeverage.sol SHA3: 4177bd110d85a43b6aebc719e7782dad5dcd7d496916672a9358437c2a2279e0 File: contracts/libs/trader-pool/TraderPoolModify.sol SHA3: 0774660e8a2801632b9e0a95aea532d82b05c28bafb1f1b9615593072e429079 File: contracts/libs/trader-pool/TraderPoolPrice.sol SHA3: aa63a767550e26db53298239f3f2df7ef7061f57acaf1c323337f7005029c0f3 File: contracts/libs/trader-pool/TraderPoolView.sol SHA3: adc624a80827a9017040929bd3a2edf6334e1ec9ac77c2a037c5a5cbb1f97d3e File: contracts/libs/trader-pool-proposal/TraderPoolInvestProposalView.sol SHA3: c6bfc85f2fd3531fd06891cf3f96e0eaeb36a5c17519d1b58a05812d73c8f4d0 File: contracts/libs/trader-pool-proposal/TraderPoolRiskyProposalView.sol SHA3: eb0af6263caaefba553002dd197c500c5077c6b88d580565b0c260ab9f2fec58 File: contracts/libs/utils/AddressSetHelper.sol SHA3: 1075d40282f45bf5405709694dfbcdc76d073e3334fc2d69ff2fce713a9243c0 File: contracts/libs/utils/DataHelper.sol SHA3: 78204bdb3cfb941360492a7dffd4c4a92879a89080214fa7279b5f9d45fffccd File: contracts/libs/utils/TokenBalance.sol SHA3: 5d618e4e842a42325ee4151f163cb5cd97ddf3c58ea29d716236d8461b784a85 File: contracts/trader/BasicTraderPool.sol SHA3: a8e619ecf24746996e19f9ba1eef065d2515f8e1dd5a6803f474490e5fc8b191 File: contracts/trader/InvestTraderPool.sol SHA3: 44726c606bfdb4eafcc96ed39ec4fe33043df70b36e62e83dc082c9721445362 File: contracts/trader/TraderPool.sol SHA3: 3ca25c0d824cc974a0ff9414b7143c391015baed62e1e75724ada9f7c96c28ae File: contracts/trader/TraderPoolInvestProposal.sol SHA3: 7c0797ddce53469659eb4bfbcd819ac1530757c0fb84bc68b61e2e2b6de353a9 File: contracts/trader/TraderPoolProposal.sol SHA3: ab30267a8ac8d078d8e40a56fcbf2d0421d3ecf87593ebafb43699f4d8517e8b File: contracts/trader/TraderPoolRiskyProposal.sol SHA3: 79a26b4bad51f46fb4d73da939250ca06a6f363a87611865b826800de32b0af3 File: contracts/user/UserRegistry.sol SHA3: a5b4ff5e7199214a1be67ff8201f83792b2638013594257087ec276669937503

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Severity Definitions

Risk Level	Description
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to the loss of user funds or contract state manipulation by external or internal actors.
High	High vulnerabilities are usually harder to exploit, requiring specific conditions, or have a more limited scope, but can still lead to the loss of user funds or contract state manipulation by external or internal actors.
Medium	Medium vulnerabilities are usually limited to state manipulations but cannot lead to asset loss. Major deviations from best practices are also in this category.
Low	Low vulnerabilities are related to outdated and unused code or minor gas optimization. These issues won't have a significant impact on code execution but affect code quality

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Executive Summary

The score measurement details can be found in the corresponding section of the scoring methodology.

Documentation quality

The total Documentation Quality score is **10** out of **10**.

- Provided documentation well describes the project.

Code quality

The total Code Quality score is **10** out of **10**.

- Code is well written, with scalable architecture and divided with components that follow the single responsibility principle.
- Missing events for some critical state changes are found.

Test coverage

Code coverage of the project is **100%** (branch coverage).

Security score

As a result of the audit, the code contains **3** low severity issues. The security score is **10** out of **10**.

All found issues are displayed in the “Findings” section.

Summary

According to the assessment, the Customer's smart contract has the following score: **10.0**.

Table. The distribution of issues during the audit

Review date	Low	Medium	High	Critical
8 March 2023	12	9	14	2
12 April 2023	4	2	0	0
22 May 2023	4	1	0	0
22 Jun 2023	3	0	0	0

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Risks

- The system is composed of multiple upgradable contracts. In case of owner keys leak, unauthorized users may receive access to user funds.
- *TraderPoolInvestProposal* allows traders to withdraw invested funds to traders' wallets. There are no guarantees that investors would take their funds back.
- *TraderPoolRiskyProposal* enables trading of custom tokens. Potential fraud may happen through the creation and manipulation of malware coins. This may result in user investment loss during trading in risky pools.
- Users may receive no insurance claims. Decisions on paying insurance are done off-chain or on with out-of-scope contracts.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

System Overview

DeXe Network is a platform for Defi asset management.
It supports the following features for users:

- create a personalized trading token,
- copy other wallets,
- save and multiply assets by investing in existing traders on the platform.

The system consists of the following contracts:

- *ContractRegistry* - a contract that stores addresses of all other DeXe contracts.
- *PoolFactory* - a contract that allows deploying trading and government pools.
- *CoreProperties* - a storage contract for shared properties between DeXe contracts.
- *PriceFeed* - Uniswap wrapper to exchange tokens and receive their price.
- *Insurance* - a contract that allows buying insurance and that manages insurance claims (off-chain).
- *UserRegistry* - a storage contract for users' info.
- *TraderPool* - an abstract pool that implements basic trading pools logic, like investment, divestment, and trading. After investment mints LP tokens.
- *BasicTraderPool* - implementation of TraderPool that allows the creation of *RiskyProposal*.
  RiskyProposal - a sub-pool of parent pool that has its own LP tokens. To create a *RiskyProposal* - the trader should specify a token for the proposal pool.
- *InvestTraderPool* - implementation of TraderPool that allows the creation of proposals for different types of investments that are out of contract’s control (like off-chain investments).
- *TraderPoolInvestProposal* - a contract that handles the logic of proposals created by *InvestTraderPool.*
- *TraderPoolRiskyProposal* - a contract that handles the logic of proposals created by *BasicTraderPool.*
- *GovUserKeeper* - a vault storing information about users, their ERC20 tokens and NFTs.
- *GovPool* - voting contract which stores proposals, votes for them and access rights.
- *ContractsRegistry* - holds string constants.
- *TokenSaleProposal* - implementation of *ITokenSaleProposal* that acts as a class for proposals of token sales in the system.
- *GovUserKeeperLocal* - a library containing two validation methods.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

- *GovUserKeeperView* - a contract which acts as a proxy to view the state on *GovUserKeeper*.
- *GovPoolCommission* - a contract to pay commission to commission receivers.
- *GovPoolExecute* - a contract to execute proposals.
- *GovPoolRewards* - a contract to provide rewards payments.
- *ShrinkableArray* - a custom data structure, an array which can shrink in size.
- *GovPoolStaking* - a contract to act as a Front-end for staking and updating rewards in the Micropools.
- *GovPoolUnlock* - a contract with a single function unlockInProposals() which unlocks proposals if conditions are met.
- *GovPoolView* - a contract which provides several view functions to view withdrawable assets and get users' proposals.
- *GovPoolVote* - a contract which acts as an external Front-end to make votes in *GovPool*.
- *GovPoolOffchain* - a contract to trigger saving results off-chain.
- *DataHelper* - a helper contract with two functions returning blockchain-related programmatic values.
- *AddressSetHelper* - a helper contract which adds addresses to an address array.
- *MathHelper* - a helper contract with two Math helper methods.
- *GovSettings* - a contract to keep arrays of settings for various executors.
- *ERC20Sale* - a governance *ERC20* token.
- *GovValidators* - a pool of validators and proposals.
- *GovValidatorsToken* - an *ERC20* governance token.
- *ERC721Multiplier* - an *ERC721* token with multiplying and rewards functionalities.
- *ERC721Power* - an extended version of an *ERC721* token.
- *DistributionProposal* - a type of proposal representing token distribution.
- *PoolRegistry* - a contract which acts as a registry of pools.
- *GovTokenSaleDeployer* - a library to deploy a token sale via proxy.

Privileged roles

- The CoreProperties contract has the following privileged roles:
  - Owners can:
    - Add/remove whitelist tokens
    - Add/remove blacklist tokens
    - Set the maximum pool investors number
    - Set the maximum open positions number
    - Set trader leverage params

        - Set commission init and duration times
        - Set DeXe commissions percentages
        - Set the trader commission percentage
        - Set delay for risky pool
        - Set insurance parameters
        - Set gov vote limit
- The *PoolRegistry* contract has the following privileged roles:
    - Pool Factory can:
        - Associate users with a pool
        - Add a proxy pool
- The *Insurance* contract has the following privileged roles:
    - Owners can:
        - Accept claims
- The *TraderPool* contract has the following privileged roles:
    - Trader admins can:
        - Modify admins
        - Modify private investors
        - Change pool parameters
        - Reinvest commissions
        - Exchange
    - BABT holders can:
        - Invest
        - Divest
    - Trader pool contract can:
        - Mint and burn LP tokens
        - Update users' balances
- The *BasicTraderPool* contract has the following privileged roles:
    - Traders can:
        - Create proposals
        - Exchange
    - BABT holders can:
        - Invest in proposals
        - Reinvest from proposals
    - Proposal pool can:
        - Add/remove users from investors
- The *InvestTraderPool* contract has the following privileged roles:
    - Traders can:
        - Create proposals
        - Exchange
        - Invest after investment delay
    - BABT holders can:
        - Invest in proposals
        - Reinvest from proposals

    - Proposal pool can:
        - Add/remove users from investors
- The *TraderPoolInvestProposal* contract has the following privileged roles:
    - Trader admins can:
        - Change proposal restrictions
        - Withdraw funds from proposals to traders' wallet
        - Supply proposals with traders' own funds
        - Convert investment base to dividents
    - Parent trader pool can:
        - Create proposals
        - Invest into proposals
        - Divest from proposals
- The *GovValidators* contract has no directly-specified privileged roles, but only addresses holding *govValidatorsToken* tokens can vote for proposals in it.
- The *ContractsRegistry* contract is inherited from OwnableContractsRegistry, but has no mutating functionality or any role-based privileges.
- The *GovSettings* contract has the following privileged role:
    - Owners can:
        - Edit settings
        - Change executors
        - Add settings
- The *ERC20Sale* contract has the following privileged role:
    - Governance can:
        - Mint
        - Burn
        - Pause
        - Unpause
- The *GovPool* contract has the following privileged role:
    - BABTHolders can:
        - Vote for proposals
        - Deposit NFTs
        - Withdraw NFTs
        - Delegate NFTs to someone
        - Undelegate
        - Unlock user's proposals
        - Execute proposals
        - Claim rewards
        - Save off-chain results
- The *GovUserKeeper* contract has the following privileged role:
    - Owner can:

        - Deposit tokens
        - Withdraw tokens
        - Delegate tokens
        - Undelegate tokens
        - Deposit NFTs
        - Withdraw NFTs
        - Delegate NFTs
        - Undelegate NFTs
        - Create NFT Power Snapshots
        - Update max amount of locked tokens
        - Lock tokens
        - Unlock tokens
        - Lock NFTs
        - Unlock NFTs
        - Update NFT powers
        - Set ERC20 address
        - Set ERC721 address
- The *GovValidators* contract has the following privileged roles:
    - Owners can:
        - Create an external proposal
    - Holders of Governance tokens can:
        - Create an internal proposal
        - Vote
- The *ERC721Multiplier* contract has the following privileged role:
    - Owners can:
        - Mint NFTs
        - Set base URI
- The *ERC721Power* contract has the following privileged role:
    - Owners can:
        - Set NFT max power
        - Set required collateral for an NFT
        - Mint NFT
        - Set base URI
- The *DistributionProposal* contract has the following privileged role:
    - Governors contract can:
        - Execute proposals
- The *TokenSaleProposal* contract has the following privileged role:
    - Governors can:
        - Add to whitelist
        - Create a tier
        - Turn off a specific tier info view
- The *PoolRegistry* contract has the following privileged role:
    - Pool factory can:

        - Add a proxy pool
        - Associate user with a pool
- The *Insurance* contract has the following privileged role:
    - Owners can:
        - Accept insurance claim

Recommendations

- With BasicTraderPool.investProposal, the access control is `onlyBABTHolder`, with GovPool, the createProposal function access control is `onlyBABTHolder`. The `onlyBABTHolder` access control might not be sufficient for security. Following the Check Effects Interaction Pattern or using ReentrancyGuard is recommended in these cases.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Checked Items

We have audited the Customers' smart contracts for commonly known and specific vulnerabilities. Here are some items considered:

Item	Type	Description	Status
Default Visibility	SWC-100 SWC-108	Functions and state variables visibility should be set explicitly. Visibility levels should be specified consciously.	Passed
Integer Overflow and Underflow	SWC-101	If unchecked math is used, all math operations should be safe from overflows and underflows.	Passed
Outdated Compiler Version	SWC-102	It is recommended to use a recent version of the Solidity compiler.	Passed
Floating Pragma	SWC-103	Contracts should be deployed with the same compiler version and flags that they have been tested thoroughly.	Failed
Unchecked Call Return Value	SWC-104	The return value of a message call should be checked.	Passed
Access Control & Authorization	CWE-284	Ownership takeover should not be possible. All crucial functions should be protected. Users could not affect data that belongs to other users.	Passed
SELFDESTRUCT Instruction	SWC-106	The contract should not be self-destructible while it has funds belonging to users.	Not Relevant
Check-Effect-Interaction	SWC-107	Check-Effect-Interaction pattern should be followed if the code performs ANY external call.	Passed
Assert Violation	SWC-110	Properly functioning code should never reach a failing assert statement.	Passed
Deprecated Solidity Functions	SWC-111	Deprecated built-in functions should never be used.	Passed
Delegatecall to Untrusted Callee	SWC-112	Delegatecalls should only be allowed to trusted addresses.	Not Relevant
DoS (Denial of Service)	SWC-113 SWC-128	Execution of the code should never be blocked by a specific contract state unless required.	Passed
Race Conditions	SWC-114	Race Conditions and Transactions Order Dependency should not be possible.	Passed

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Authorization through tx.origin	SWC-115	tx.origin should not be used for authorization.	Passed
Block values as a proxy for time	SWC-116	Block numbers should not be used for time calculations.	Passed
Signature Unique Id	SWC-117 SWC-121 SWC-122 EIP-155 EIP-712	Signed messages should always have a unique id. A transaction hash should not be used as a unique id. Chain identifiers should always be used. All parameters from the signature should be used in signer recovery. EIP-712 should be followed during a signer verification.	Not Relevant
Shadowing State Variable	SWC-119	State variables should not be shadowed.	Failed
Weak Sources of Randomness	SWC-120	Random values should never be generated from Chain Attributes or be predictable.	Not Relevant
Incorrect Inheritance Order	SWC-125	When inheriting multiple contracts, especially if they have identical functions, a developer should carefully specify inheritance in the correct order.	Passed
Calls Only to Trusted Addresses	EEA-Level-2 SWC-126	All external calls should be performed only to trusted addresses.	Not Relevant
Presence of Unused Variables	SWC-131	The code should not contain unused variables if this is not justified by design.	Passed
EIP Standards Violation	EIP	EIP standards should not be violated.	Not Relevant
Assets Integrity	**Custom**	Funds are protected and cannot be withdrawn without proper permissions or be locked on the contract.	Passed
User Balances Manipulation	**Custom**	Contract owners or any other third party should not be able to access funds belonging to users.	Passed
Data Consistency	**Custom**	Smart contract data should be consistent all over the data flow.	Passed
Flashloan Attack	**Custom**	When working with exchange rates, they should be received from a trusted source and not be vulnerable to short-term rate changes that can be achieved by using flash loans. Oracles should be used.	Passed

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Token Supply Manipulation	**Custom**	Tokens can be minted only according to rules specified in a whitepaper or any other documentation provided by the Customer.	Passed
Gas Limit and Loops	**Custom**	Transaction execution costs should not depend dramatically on the amount of data stored on the contract. There should not be any cases when execution fails due to the block Gas limit.	Passed
Style Guide Violation	**Custom**	Style guides and best practices should be followed.	Passed
Requirements Compliance	**Custom**	The code should be compliant with the requirements provided by the Customer.	Passed
Environment Consistency	**Custom**	The project should contain a configured development environment with a comprehensive description of how to compile, build and deploy the code.	Passed
Secure Oracles Usage	**Custom**	The code should have the ability to pause specific data feeds that it relies on. This should be done to protect a contract from compromised oracles.	Passed
Tests Coverage	**Custom**	The code should be covered with unit tests. Test coverage should be sufficient, with both negative and positive cases covered. Usage of contracts by multiple users should be tested.	Passed
Stable Imports	**Custom**	The code should not reference draft contracts, which may be changed in the future.	Passed

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Findings

■■■■ Critical

C01. Unverified Interaction

The DeXe system allows traders to add any ERC20 token. Unsafe interaction with untrusted tokens could lead to unexpected contract behavior.

It is possible for a trader to implement and provide a token with the possibility to change decimals at any period in time. As the DeXe system relies on decimals during price calculation (*priceFeed.getNormalizedPriceOut* and *priceFeed.getNormalizedPriceIn*), internal contract logic may be corrupted.

This may lead to the trader manipulating internal contracts logic: investing ignoring max price, manipulating balances during investment, etc.

Path:
./contracts/trader/TraderPoolRiskyProposal.sol: invest()

Recommendation: Either a whitelisting mechanism for ERC20 tokens can be added which would be in control of an admin role, or the *decimals()* call return value of every ERC20 token could be written to storage to be used instead of calling the *decimals()* function. Another option would be to limit Gas for *decimals()* calls and make this limit configurable by the owner (to prevent any potential issue with different blockchains or future releases).

Found in: f29a0e26ae59ffdf34f62e69ae1c32c606d10fd3

Status: Mitigated (The outcome of manipulating the decimals of a token has very little effect and it does not lead to funds losses.)

C02. Flashloan Attack; Front-Running Attack

The library uses Uniswap's *getAmountsOut* and *getAmountsIn* functions to determine the exchange rate for the assets. Those functions provide the price based on the current state of a liquidity pool that may be easily manipulated by flasholans.

As a result, the *PriceFeed.sol* contract may operate with manipulated data and perform exchanges with undesirable rates.

Paths:
./contracts/libs/price-feed/UniswapV2PathFinder.sol:
getUniV2PathWithPriceOut()
./contracts/libs/price-feed/UniswapV2PathFinder.sol:
getUniV2PathWithPriceIn()

Recommendation: Fetch data from an oracle.

Found in: f29a0e26ae59ffdf34f62e69ae1c32c606d10fd3

Status: Mitigated (The project uses an off-chain minAmountsOut parameter that should protect against funds losses; however, the calculations are out of scope for the audit and remain as a risk.)

■■■ High

H01. Requirements Violation

In a risky pool, traders can trade any token, even those created by them, and add liquidity to Uniswap.

This allows the trader to always be in profit by selling their tokens at a higher price, and closing DeXe trades at a loss.

Path:
./contracts/trader/TraderPoolRiskyProposal.sol: create()

Recommendation: Consider limiting tokens for the risky pool using the first top X tokens from CMC.

Found in: f29a0e26ae59ffdf34f62e69ae1c32c606d10fd3

Status: Mitigated (This behavior is accepted as the business logic of Risky Trader Pools)

H02. Highly Permissive Role Access

The owner of the contract could change commissions at any period of time. The upper bound limit for the commission is not set up.

This may lead to the manipulation of fees.

Path:
./contracts/core/CoreProperties.sol: setDEXECommissionPercentages(), setTraderCommissionPercentages()

Recommendation: Hardcode the upper bound for each commission and disallow set fees higher than the bound value.

Found in: f29a0e26ae59ffdf34f62e69ae1c32c606d10fd3

Status: Fixed (Revised commit: 24ce9a6)

H03. Requirements Violation

According to the provided requirements, the *Insurance* contract should have the *proposeClaim()* function. The function's purpose is to allow users to open disputes in insurance cases.

However, the function is not provided. The user interaction flow is not described.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

According to the documentation, it should be possible to view users and their deposits in an insurance pool. However, the functionality is not implemented.

Path:
./contracts/insurance/Insurance.sol: acceptClaim()

Recommendation: Implement code according to requirements.

Found in: f29a0e26ae59ffdf34f62e69ae1c32c606d10fd3

Status: Mitigated (Intended logic. The proposeClaim() function is the createProposal() function on DEXE DAO)

H04. Assets Integrity; Undocumented Behavior

TraderPoolInvestProposal allows traders to withdraw all user's investments from the pool. There is no guarantee that users would be able to receive their funds back.

The logic critically influences users. It should be highlighted in the publicly available documentation, so users are aware of the potential losses and manipulations.

Path:
./contracts/trader/TraderPoolInvestProposal.sol: withdraw()

Recommendation: Highlight this behavior in the publicly available documentation.

Found in: f29a0e26ae59ffdf34f62e69ae1c32c606d10fd3

Status: Mitigated (From provided doc: “Investment fund allows traders to WITHDRAW ALL user's investments from the pool. There is NO guarantee that users would be able to receive their funds back. Invest only in funds you can trust.”)

H05. Highly Permissive Role Access

The owner of the contract may burn any user funds.

User funds should not be accessible without proper allowances.

Path:
./contracts/gov/ERC20/ERC20Sale.sol: burn()

Recommendation: Use the *ERC20Burnable* pattern if a burnable functionality is needed.

Found in: f29a0e26ae59ffdf34f62e69ae1c32c606d10fd3

Status: Fixed (Revised commit: 24ce9a6)

H06. Denial of Service Vulnerability

The voting duration should be greater than a minimal value.

This may lead to not reaching a quorum as the proposal quickly comes to a *Defeated* state.

Path:
./contracts/gov/validators/GovValidators.sol: duration

Recommendation: Provide a reasonable minimal value for the duration variable.

Found in: f29a0e26ae59ffdf34f62e69ae1c32c606d10fd3

Status: Mitigated (The client has confirmed, that it is by design the responsibility of the users to configure their DAO correctly)

H07. Highly Permissive Role Access

The owner of the contract may change validators at any moment.

The functionality is considered to be highly permissive as the voting process is totally under owner control.

Path:
./contracts/gov/validators/GovValidators.sol: changeBalances()

Recommendation: Provide documentation describing why the functionality is needed or remove the function.

Found in: f29a0e26ae59ffdf34f62e69ae1c32c606d10fd3

Status: Fixed (Revised commit: 24ce9a6)

H08. Integer Overflow

The function *_setERC721Address* has the *require(nftsTotalSupply > 0)* check implemented. However, due to downcasting of *nftsTotalSupply* to *uint128*, this condition could be bypassed.

It happens when casting from a number which has the first 128 bits as zeros.

This may lead to an unexpected *nftsTotalSupply* value assigned.

Path:
./contracts/gov/user-keeper/GovUserKeeper.sol: _setERC721Address()

Recommendation: Upcast *_nftInfo.totalSupply* to *uint256* or add safeguards against unsafe casting.

Found in: f29a0e26ae59ffdf34f62e69ae1c32c606d10fd3

Status: Fixed (Revised commit: 24ce9a6)

H09. Denial of Service Vulnerability

In case of setting voting duration close to the *uint64* max value, *voteEnd* voting field may be overflowed and newly created votings can come to a *Defeated* state.

This may lead to an inability to reach quorum and execute any voting.

Path:
./contracts/gov/validators/GovValidators.sol: createExternalProposal(), createInternalProposal()
./contracts/libs/gov-pool/GovPoolCreate.sol: createProposal()

Recommendation: Bound voting duration from top to make *uint64(block.timestamp + duration)* never overflow.

Found in: f29a0e26ae59ffdf34f62e69ae1c32c606d10fd3

Status: Mitigated (The issue is mitigated after being recognized after a meeting with the client as posing no sizeable economic impact to users)

H10. Highly Permissive Role Access

The injector is able to change the DeXe token address on the contract.

This may lead to users being unable to withdraw their deposits and the injector may take control over the insurance pool.

Path:
./contracts/insurance/Insurance.sol: setDependencies()

Recommendation: Remove the ability to update the *_dexe* variable on the contract.

Found in: f29a0e26ae59ffdf34f62e69ae1c32c606d10fd3

Status: Fixed (Revised commit: 24ce9a6)

H11. Denial of Service Vulnerability

Each new proposal creates a new structure with *snapshotId* as its unique identifier. *snapshotId* is capped by a *uint32* type, which has a max value of *4294967295*. The potential attacker may max out this value in the event of a potential gas price drop.

This may lead to the use of outdated snapshots and retired validators overtaking the quorum.

Path:
./contracts/gov/validators/GovValidators.sol: createInternalProposal()

Recommendation: Upcast *snapshotId* to a higher type or provide another safeguard mechanism.

Found in: f29a0e26ae59ffdf34f62e69ae1c32c606d10fd3

Status: Fixed (Revised commit: 24ce9a6)

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

H12. Access Control Violation; Race Conditions

The purpose of the *setDependencies()* function is to set up critical parameters and addresses. It is equipped with the *dependant* modifier.

The *dependant* modifier works as following:

- if the injector role is not yet set, sets the injector as the caller and allows the transaction
- if the injector is set and it is the caller, allows the transaction
- if the injector is set and it is not the caller, rejects the transaction.

Since, for the first time, the setter functions could be called by anyone, there is a risk of race conditions that may result in the inconsistent state of the contract.

For example, an attacker may call the function right after the contract is deployed and set himself as the injector.

This may lead to access to critical state variables by an unauthorized user.

Paths:
./contracts/core/CoreProperties.sol: setDependencies()
./contracts/core/PriceFeed.sol: setDependencies()
./contracts/gov/GovPool.sol: setDependencies()
./contracts/insurance/Insurance.sol: setDependencies()
./contracts/trader/BasicTraderPool.sol: setDependencies()
./contracts/trader/InvestTraderPool.sol: setDependencies()
./contracts/trader/TraderPool.sol: setDependencies()
./contracts/trader/TraderPoolProposal.sol: setDependencies()

Recommendation: Call the *setDependencies()* function in the initializer functions of the corresponding contracts.

Found in: f29a0e26ae59ffdf34f62e69ae1c32c606d10fd3

Status: Mitigated (The *setDependencies()* call is made just after deployment and in the worst case scenario where an attacker accesses the state variables, the contracts can be redeployed before any funds are accepted)

H13. Upgradeability Issues

The contracts are abstract and upgradable but do not follow the upgradability best practices by not adding a gap in the contract storage.

This may lead to a child contract storage layout corruption during an upgrade.

Paths:
./contracts/trader/TraderPool.sol
./contracts/trader/TraderPoolProposal.sol

Recommendation: Add a gap to the contract storage to allow future upgradability.

Found in: f29a0e26ae59ffdf34f62e69ae1c32c606d10fd3

Status: Fixed (Revised commit: 24ce9a6)

H14. Requirements Violation; Misleading Naming

According to the naming, the contract should provide an insurance functionality. However, it provides only an ability to lock funds and then withdraw them.

The *buyInsurance* function does not reflect exactly what happens with the user funds.

This may lead to wrong assumptions about the functionality's purpose.

Path:
./contracts/insurance/Insurance.sol

Recommendation: Provide declarative names which correspond to the code's purpose, provide publicly available documentation for the functionality.

Found in: f29a0e26ae59ffdf34f62e69ae1c32c606d10fd3

Status: Mitigated (The functionality and naming are not changed; however, there is documentation stating that the functionality is only to lock tokens that correspond to the insurance values and the actual insurance process is determined by the DEXE DAO.)

Medium

M01. Best Practice Violation - CEI Pattern Violation

The Checks-Effects-Interactions pattern is violated. In some functions, the state variables are modified after doing external calls, which is against best practices.

- In the *TraderPoolInvestProposal.sol* *create()* function the *_proposalInfos* state variable is updated after the external call to the *_parentTraderPoolInfo.baseToken* contract.
- In the *TraderPoolInvestProposal.sol* *invest()* function the *info* state variable is updated after the external call to the *_parentTraderPoolInfo.baseToken* contract.
- In the *TraderPoolRiskyProposal.sol* the *create()* function *_proposalInfos* state variable is updated after the external call to the *priceFeed* contract.
- In the *TraderPoolRiskyProposal.sol* *invest()* function the *info* state variable is updated after the external call to the *priceFeed* contract.
- In the *TraderPoolRiskyProposal.sol* *_investActivePortfolio* function some variables are modified after the external call to the the *priceFeed* contract.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

These may lead to reentrancies, race conditions, and denial of service vulnerabilities during the implementation of new functionality.

Paths:
./contracts/trader/TraderPoolInvestProposal.sol: create(), invest()
./contracts/trader/TraderPoolRiskyProposal.sol: create(), invest(), _investActivePortfolio()

Recommendation: Update state variables before an external call or use ReentrancyGuard instead. Since some of the modifications of the parameters depend on the return values of the external calls, the Check-Effects-Interaction pattern cannot be followed. The ReentrancyGuard module from OpenZeppelin can be used in those cases.

Found in: f29a0e26ae59ffdf34f62e69ae1c32c606d10fd3

Status: Fixed (Revised commit: bd1044e)

M02. Inconsistent Data - Unused Return Value

Multiple functions perform transfer calls on the *_dexe* token but ignore the return value.

Path:
./contracts/insurance/Insurance.sol: buyInsurance(), _payout(), withdraw()

Recommendation: Use *SafeTransfer* to check if transfers are successful.

Found in: f29a0e26ae59ffdf34f62e69ae1c32c606d10fd3

Status: Mitigated (Since the calls are made to the trusted DEXE token the issue can be mitigated.)

M04. Best Practice Violation - Unfinalized Functionality

It is not possible to execute an external proposal even if a quorum is reached.

This may lead to double execution of the proposal off-chain as the proposal state could not be updated.

Path:
./contracts/gov/validators/GovValidators.sol: createExternalProposal()

Recommendation: Finalize the contract to make it possible to execute external proposals.

Found in: f29a0e26ae59ffdf34f62e69ae1c32c606d10fd3

Status: Fixed (Revised commit: 24ce9a6)

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

M05. Contradiction - Denial of Service

The *onlyValidator* modifier uses current user balances to identify validators. However, voting is only possible for those who have balances at the moment of proposal creation.

This may lead to an inability to reach a quorum on proposals due to some validators not passing the modifier.

Path:
./contracts/gov/validators/GovValidators.sol: onlyValidator()

Recommendation: Check if the user was a validator based on a specified snapshot.

Found in: f29a0e26ae59ffdf34f62e69ae1c32c606d10fd3

Status: Fixed (Revised commit: 24ce9a6)

M07. Contradiction - Invalid Return Action

The function contains the check *(bytes(poolName).length == 0)* which returns *address(0)*. This might cause a logic error, because the relying code might not expect a zero address. Instead, an error should be thrown clearly marking that the input parameter is invalid.

Path:
./contracts/factory/PoolFactory.sol: predictGovAddress()

Recommendation: Replace returning *address(0)* with *revert()* or *require()*.

Found in: f29a0e26ae59ffdf34f62e69ae1c32c606d10fd3

Status: Mitigated (Since the function is declared as view, reverting is not necessary)

M08. Inconsistent Data - Sign of Unfinalized Code

The *_validateGovPoolWithTokenSaleParameters* function contains the check *require(parameters. ... Executors[0] == address(0))*.

If the value should always be equal to *0x0* it may be directly overwritten there.

The validate function is only used by the *deployGovPoolWithTokenSale* function. However, there are other functions which receive the *GovPoolDeployParams* structure as parameters.

Path:
./contracts/factory/PoolFactory.sol:
_validateGovPoolWithTokenSaleParameters()

Recommendation: Provide proper NatSpec for the functions, clarify the behavior.

Found in: f29a0e26ae59ffdf34f62e69ae1c32c606d10fd3

Status: Fixed (Revised commit: 24ce9a6)

M09. Best Practice Violation - Unstable Import

It is against best practices to deploy deterministic code, which was not tested thoroughly.

The project allows different versions of a custom import `@dlsl/dev-modules`: `^1.8.1`.

This may lead to unexpected and untested code being deployed.

Path:
./package.json: @dlsl/dev-modules

Recommendation: Fix the module version to `1.8.1` or `1.9.0`.

Found in: f29a0e26ae59ffdf34f62e69ae1c32c606d10fd3

Status: Fixed (Revised commit: 24ce9a6)

Low

L01. Floating Pragma

Locking the pragma helps ensure that contracts do not accidentally get deployed using, for example, an outdated compiler version that might introduce bugs that affect the contract system negatively.

Path: All files

Recommendation: Lock the compiler version.

Found in: f29a0e26ae59ffdf34f62e69ae1c32c606d10fd3

Status: Reported (The pragma is not locked on the contract level. Check Link for more information.)

L02. Redundant Import

Unused imports should be removed from the contracts. Unused imports are allowed in Solidity and do not pose a direct security issue. However, it is best practice to avoid them as they can decrease readability.

Path:
./contracts/factory/PoolRegistry.sol: Math

Recommendation: Remove unused import.

Found in: f29a0e26ae59ffdf34f62e69ae1c32c606d10fd3

Status: Reported (Import removed, but the `using` keyword remains.)

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

L03. Missing Events

Events for critical state changes should be emitted for tracking things off-chain.

Paths:
./contracts/core/PriceFeed.sol: addPathTokens(), removePathTokens()
./contract/core/CoreProperties.sol: all set functions

Recommendation: Emit event after changes in the contract.

Found in: f29a0e26ae59ffdf34f62e69ae1c32c606d10fd3

Status: Reported (Critical state changes do not emit any events.)

L04. Spelling Error

The *require* check message contains misspellings. The word *percentage* is misspelled as *percantage*.

Path:
./contracts/trader/TraderPoolRiskyProposal.sol: create()

Recommendation: Change *percantage* to *percentage*.

Found in: f29a0e26ae59ffdf34f62e69ae1c32c606d10fd3

Status: Fixed (Revised commit: 24ce9a6)

L05. Unindexed Events

Having indexed parameters in the events makes it easier to search for these events using indexed parameters as filters.

Paths:
./contracts/factory/PoolFactory.sol
./contracts/trader/TraderPool.sol

Recommendation: Use the *indexed* keyword for the event parameters.

Found in: f29a0e26ae59ffdf34f62e69ae1c32c606d10fd3

Status: Mitigated (Indexed parameters are not needed right now since the backend of the system relies on events)

L06. Function Which May Be Declared Private

There is no added value in declaring non-public functions as *internal* in a non-inherited contract.

Path:
./contracts/factory/PoolFactory.sol

Recommendation: Replace *internal* visibility modifiers with *private* in non-inheritable contracts.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Found in: f29a0e26ae59ffdf34f62e69ae1c32c606d10fd3

Status: Mitigated (The internal visibilities are not changed in case inheritance is introduced)

L08. Style Guide Violation

The names *lastvalue_* and *DP* violate Solidity Style Guide conventions, which prefer camel case for variable names.

Paths:
@dlsl/dev-modules/libs/data-structures/StringSet.sol: remove()
./contracts/factory/PoolFactory.sol: DaoPoolDeployed.DP

Recommendation: Rename the variable *lastvalue_* to *lastValue_* and *DaoPoolDeployed.DP* to *DaoPoolDeployed.dp*.

Found in: f29a0e26ae59ffdf34f62e69ae1c32c606d10fd3

Status: Fixed (Revised commit: 24ce9a6)

L09. Redundant Statement

The *onlyValidator* check on the functions is redundant as the functions call *_beforeTokenTransfer* functions internally, which is under the modifier.

Path:
./contracts/gov/validators/GovValidatorsToken.sol: mint(), burn()

Recommendation: Remove redundant modifiers.

Found in: f29a0e26ae59ffdf34f62e69ae1c32c606d10fd3

Status: Fixed (Revised commit: 24ce9a6)

L10. Missing Zero Address Validation

Address parameters are used without checking against the possibility of *0x0*.

This may lead to unwanted external calls to *0x0*.

Paths:
./contracts/gov/GovPool.sol: __GovPool_init()
./contracts/gov/settings/GovSettings.sol: __GovSettings_init()
./contracts/trader/BasicTraderPool.sol: __BasicTraderPool_init()
./contracts/trader/InvestTraderPool.sol: __InvestTraderPool_init()

Recommendation: Implement zero address checks.

Found in: f29a0e26ae59ffdf34f62e69ae1c32c606d10fd3

Status: Mitigated (It is confirmed that the deployment is done using factory contracts and that there is no risk of zero addresses.)

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

L12. Shadowing State Variables

State variables should not be shadowed in order to keep abstraction levels clear.

In the *__TraderPool_init()* function and constructors, there are *name* and *symbol* parameters. The parent Smart Contract *ERC20Upgradeable.sol* already has smart contract members with the same names.

Paths:
./contracts/trader/TraderPool.sol: __TraderPool_init()
./contracts/trader/InvestTraderPool.sol: __InvestTraderPool_init()
./contracts/trader/BasicTraderPool.sol: __BasicTraderPool_init()
./contracts/gov/validators/GovValidatorsToken.sol: constructor()
./contracts/gov/ERC721/ERC721Power.sol: constructor()
./contracts/gov/ERC721/ERC721Multiplier.sol: constructor()

Recommendation: Rename the *name* and *symbol* parameters e.g. adding an underscore.

Found in: f29a0e26ae59ffdf34f62e69ae1c32c606d10fd3

Status: Mitigated (This is an informational issue and does not lead to vulnerabilities.)

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Disclaimers

Hacken Disclaimer

The smart contracts given for audit have been analyzed based on best industry practices at the time of the writing of this report, with cybersecurity vulnerabilities and issues in smart contract source code, the details of which are disclosed in this report (Source Code); the Source Code compilation, deployment, and functionality (performing the intended functions).

The report contains no statements or warranties on the identification of all vulnerabilities and security of the code. The report covers the code submitted and reviewed, so it may not be relevant after any modifications. Do not consider this report as a final and sufficient assessment regarding the utility and safety of the code, bug-free status, or any other contract statements.

While we have done our best in conducting the analysis and producing this report, it is important to note that you should not rely on this report only — we recommend proceeding with several independent audits and a public bug bounty program to ensure the security of smart contracts.

English is the original language of the report. The Consultant is not responsible for the correctness of the translated versions.

Technical Disclaimer

Smart contracts are deployed and executed on a blockchain platform. The platform, its programming language, and other software related to the smart contract can have vulnerabilities that can lead to hacks. Thus, the Consultant cannot guarantee the explicit security of the audited smart contracts.