HACKEN

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Customer: DAC Portal
Date: October 5th, 2022

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

This report may contain confidential information about IT systems and the intellectual property of the Customer, as well as information about potential vulnerabilities and methods of their exploitation.

The report can be disclosed publicly after prior consent by another Party. Any subsequent publication of this report shall be without mandatory consent.

Document

Name	Smart Contract Code Review and Security Analysis Report for DAC Portal
Approved By	Evgeniy Bezuglyi | SC Audits Department Head at Hacken OU Noah Jelich | Senior Solidity SC Auditor at Hacken OU
Type	Staking, ERC20, Multiple Purpose System
Platform	EVM
Network	Metis
Language	Solidity
Methods	Manual Review, Automated Review, Architecture review
Website	https://www.metis.io/
Timeline	30.06.2022 - 05.10.2022
Changelog	14.07.2022 - Initial Review 29.07.2022 - Second Review 26.08.2022 - Third Review 21.09.2022 - Fourth Review 05.10.2022 - Fifth 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 DAC Portal (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 smart contracts in the repository:

Initial review scope
Repository:
https://github.com/metisdac/Metis-DAC-contracts
Commit:
1ec034fef5f8c3f40b844b5f0bed115a4c22730f
Technical Documentation:
Type: Whitepaper (partial functional requirements provided)
Link

Integration and Unit Tests: No
Deployed Contracts Addresses: No
Contracts:
File: ./contracts/AddressList.sol
SHA3: 786133bb4fd786d61a62b11c3356b2be177c99b7cfa877115d821c509c5b06c3

File: ./contracts/Configuration.sol
SHA3: 97a2f16591abf3fc39dd71e27451835714ced95618b47affd829cd3785b93297

File: ./contracts/constants.sol
SHA3: 5b352008978df3a74fa4b90f5490f9b3c32827b8d2fea52b5a678bcf58c87c92

File: ./contracts/Controller.sol
SHA3: 562cde38c2fe1b7577c06b6c77e3939fbe1ac0bef9cd0f5bdf4c501b370b7167

File: ./contracts/DAC.sol
SHA3: 14b56438062846aac0443ca53cc9a4e855c2ab30567edac8b89ec0cd27963ff6

File: ./contracts/DACFactory.sol
SHA3: 6b05979548c032d8f59692f08132171bfeca194de3dff03ebcb7a8948a984aff

File: ./contracts/DACHelper.sol
SHA3: e0ae57b6b07d7a6bb23435156f84df255ff0fbe3a49ce1df02029f8b9786da78

File: ./contracts/EcoNode.sol
SHA3: 770437016ef39632d28347c207eb160adf8c5dc3584d9c53c43ac2dfc617b819

File: ./contracts/interfaces/IController.sol
SHA3: af021b3861e8f7268fb25dd1b0716b2868e21a67dd5d645fdaa476a002f50c3a

File: ./contracts/interfaces/IDAC.sol
SHA3: 1521b224b56792705ca87c5d33cad0135b146c92d68f73f52f03b24a7170a8b0

File: ./contracts/interfaces/IDACFactory.sol
SHA3: 419412af4570f506ddd84e6077e8209215da633c5875e5210f585329ea21a53f

File: ./contracts/interfaces/IDACHelper.sol
SHA3: 19d46055a76f76809c913dd671861b28dfd492b6d3e716db985458de347df956

```
File: ./contracts/interfaces/IEcoNode.sol
SHA3: 137dfff2b103262ad0a17f6cedbe6fad47b5c179697e5c5d3b9dfe06e369b24f

File: ./contracts/interfaces/IMetisNFTBadge.sol
SHA3: e555501c43989fe6aa93ea631edb26dadff826edc25df8a5090cf93d19761593

File: ./contracts/interfaces/IMetisNFTBadgeMetadata.sol
SHA3: 7f93bcc6bf69c4ea4ba2a577aeed1eb632c8b8c1fa2df90c79043dfd8c873e87

File: ./contracts/interfaces/IMetisSocialNFT.sol
SHA3: 3aec658a9f63c11a1c7f9ac441392552501b199a5b75fdc3e6e6d0ec3f02daf3

File: ./contracts/interfaces/IProxyFactory.sol
SHA3: c43b2eef73f859a33fd5112f23f3bf584885b4e48d7be0141ba26a1ebdb99b68

File: ./contracts/interfaces/IRP.sol
SHA3: 8b404e1e056b41fca2357706c1e62d41301e51af2911c49ba592d03228fa2f6d

File: ./contracts/interfaces/ISingleton.sol
SHA3: 69d08b1ba12c41c35b876afaf17c3be134ce859687362efa6af666d592e11662

File: ./contracts/interfaces/ISingletonManager.sol
SHA3: 1f24856fa5f0b0ec9f9971fbcb09adf991b70796fac6dce70604db76ec4c9949

File: ./contracts/interfaces/safe.sol
SHA3: ca5dbaa35c5876696c0f614709edca64a3f9ee43d682b590ff303332196c08d6

File: ./contracts/MasterDeployer.sol
SHA3: ee096e65b1bbdb6ddf6bf299b32256b642a378f1da968d4cdbfbe06961948cbc

File: ./contracts/MemberManager.sol
SHA3: 16297e0420d016a6207fceac88add694536ba1114075531a9961cac157fd9fdb

File: ./contracts/Proxy.sol
SHA3: d83dc2520da768c674211ae7383ea6ff4faace788d533e13d698722b4f7c8846

File: ./contracts/ProxyFactory.sol
SHA3: 1a907c30630719683f084437996d0c151e8364f7272537b0ca55e7c02fbe5a02

File: ./contracts/Reward.sol
SHA3: dac401e1a65ccf269956be8013f7cf2fe409810540a246c7ef96636223b5614e

File: ./contracts/RP.sol
SHA3: b904d14050a9462bcb0512218be082dbb62fc5afda6f69a02a4fbb06df108e72

File: ./contracts/Singleton.sol
SHA3: d0dea87b0d24819c5277755fa315e7a7da06396614f7726bd6d8aa941aabf803

File: ./contracts/SingletonManager.sol
SHA3: 5e29562aa85b69ec015a6adc8d9cab115a58d0eb4f88757ba155ccc8730b5ca8

File: ./contracts/test/MockBadge.sol
SHA3: 2c273d81b1dac36524c202094fdee7280ccb06397b9330722020741236f27213

File: ./contracts/util.sol
SHA3: 46c9983af0e69e8aabef5d1558afe73a37c9901ace1c9e9873294b20f9661ef3

File: ./contracts/VeMetis.sol
SHA3: 35d8bfa23ed061b2e62e5ac19faa75388a6aa46c6ab5ac50bde48cb12c3d6df2
```

Second review scope

Repository:
https://github.com/metisdac/Metis-DAC-contracts

Commit:
cab2df387ea7d9b8219e7056890b1bf967a7e3ae

Technical Documentation:
Type: Whitepaper (partial functional requirements provided)
Link

Integration and Unit Tests: Yes
Deployed Contracts Addresses: No
Contracts:

File: ./contracts/AddressList.sol
SHA3: 8b40e4fae1b94c31f5d6398ff89fed1dcb7b7b711ce77123545b35110a5a808f

File: ./contracts/ConfigKeys.sol
SHA3: 5475e2cee1eee905a6e37f7dc377eb3b039925b418cc93876799cbbe8301b7f3

File: ./contracts/Configuration.sol
SHA3: 61a0238cc7f846de0316c92dd8b74dc56691f0624cbe2fc15fff51951ce446b3

File: ./contracts/constants.sol
SHA3: 3aa139212193e48f7fa6e3b73edf841bbd9f43278c2e7a535d6ea51fd3341cc4

File: ./contracts/Controller.sol
SHA3: 6889b8f37d01ad5ade3a603bed06f1bced03e216329394cdc20b19696ee4849c

File: ./contracts/DAC.sol
SHA3: 1fd4e68d6dbb2719ca8393433fa5a29fcbd267148e1b1fd92c682b8581da6ed4

File: ./contracts/DACFactory.sol
SHA3: 060f6c5aa71d7f8e3260c74d28d1d81a7662cf461217482695272cb7169f87d2

File: ./contracts/DACHelper.sol
SHA3: 81a442c975577756bc86f5535a8978347f605e5a5b939ac19ef75441ed9a6c88

File: ./contracts/EcoNode.sol
SHA3: 5c29c50269d6d23ab8e02aa21b2b1b6a44f07fdf76ecd4f944cc055a1ccffe04

File: ./contracts/ErrorCodes.sol
SHA3: 809fc3e79affa1bc367a508c4482a437cdde7171d1fe0b0a133ceaa380af003b

File: ./contracts/interfaces/IBadgeManager.sol
SHA3: f7274fa5ea1c538402bd40be09a1b67c0be58182f22590f1c79b90d51882d1fb

File: ./contracts/interfaces/IController.sol
SHA3: c881a61eb0268357a7ad46c378b3be1715c38bca514e77d39323bce0d6438f81

File: ./contracts/interfaces/IDAC.sol
SHA3: 41595535e07d104ca943b3569f57474e09103354c626d73c79b10e6fc4d5d035

File: ./contracts/interfaces/IDACFactory.sol
SHA3: 0636bb56a1be08f6cd4cf0048ac889414fd9a5a9109a17c6fac9524eac305923

File: ./contracts/interfaces/IDACHelper.sol
SHA3: 0abd27505740eb269eb78990a7ad1bf669d1d12c393ac0ad8138c5e22d957f2e

File: ./contracts/interfaces/IEcoNode.sol
SHA3: 83833321a5f5e267de3830efc35d9b68c8eb199d54a098d0d2f975b9fe97f59a

```
File: ./contracts/interfaces/IMetisNFTBadge.sol
SHA3: 04d4ab2975a2aaf571773e3299ab3d44b1852d319bcd3bc31dfc552a4eb88506

File: ./contracts/interfaces/IMetisNFTBadgeMetadata.sol
SHA3: 4144c44e324cb0cf75ba30cba6b88f20113a357af577a763b4cfff4054c82bd6

File: ./contracts/interfaces/IProposalFactory.sol
SHA3: fcf906c3558d3f86013cc1a3ef66c29caa5fbc123a8f51dbda976718119cf2b5

File: ./contracts/interfaces/IProxyFactory.sol
SHA3: 68d72c14b86104f956f1f2064df11c0986129ce8791e6a954e0943b6b8e47b40

File: ./contracts/interfaces/IRP.sol
SHA3: 2c39e024cba2ca9b54c4729b0ea7499ee38888aeff81cd1aa74eca095f673d4d

File: ./contracts/interfaces/ISimpleProposal.sol
SHA3: 4ead7576e32329c80a7b17ddd4fc83e45207f4d0ec531a614eb8f2bcabbf6093

File: ./contracts/interfaces/ISingleton.sol
SHA3: ed5aec341853cea7e3d3a717518a1a08b470095bd6b27d387051a4aa744f6e77

File: ./contracts/interfaces/ISingletonManager.sol
SHA3: 33a42fbff497f4e0bd7d2c8a22f026d1ff05af56850b73388c1a2cee4e08cd85

File: ./contracts/interfaces/safe.sol
SHA3: 763fc81b1f50be61539ab9c9f5775937af92f3c0035fb812efaafa18ab703be5

File: ./contracts/MasterCopyNames.sol
SHA3: 3165c5146208d349a295fffb4a86472fd803c3aabecb4cd542cb672cea4a98bc

File: ./contracts/MemberManager.sol
SHA3: 9eba881807314a03dede9cb672605f4add867a9514053d54b95da9f04c273064

File: ./contracts/ProposalFactory.sol
SHA3: 7239a22a3485e53a0bbc305483e85ac0325759c495a570257cca624bd382dfb3

File: ./contracts/Proxy.sol
SHA3: a06380983916e426a4eb04f0ad905cf0c8fb5b5c616c98c2f316913b39b14a9e

File: ./contracts/ProxyFactory.sol
SHA3: 5201ab81128ed17b0ef83f043f6bb1f48c8b55fbd6d36a9fe6a4cac76022010b

File: ./contracts/Reward.sol
SHA3: ce8800a86d6a4edacf50e42293a3b3e6470cbc3b8c4f5e8da6130743d3871ba9

File: ./contracts/RP.sol
SHA3: 87db317954fff018fc2fe3fcbe344c19f18a7917002c8d2c0d215f3f85a4a300

File: ./contracts/SimpleProposal.sol
SHA3: 15bb5e74d526d1f2d6758b9697d89d6a93aeadb3c5ed1e697edf03bdcdb2bf1c

File: ./contracts/Singleton.sol
SHA3: 9946a7a769fb8246f8843e53008c9e0a5e9559525fe27dbb8bcdad2a3ee16c7f

File: ./contracts/SingletonManager.sol
SHA3: b8c24035d760b48a76b6a9444d4fa4a36dd8c4e446c2f59d1c09bba0474ad463

File: ./contracts/test/MockBadge.sol
SHA3: bc3ee466d58d23c6a7bab9dcdebe392583d169ee80424ba0f2f0fe09c8ff5803
```

File: ./contracts/test/MockBadgeManager.sol
SHA3: ebc3bd1fb7f8e5e3275a769a1c984f691f2bc0479a449172ead521f30526f959

File: ./contracts/test/MockBadSafe.sol
SHA3: cc042d0eaac4ae806cc0acd97f62a7343be335bdd148675f2fe36e040080739c

File: ./contracts/test/MockToken.sol
SHA3: a4644672430ea7810aa2d3a4fd290a0eb59f17ee599e8cd8f83cf5c0c0cf86a4

File: ./contracts/util.sol
SHA3: c173ed9a7208724a07b560ad9309d52db98e7b90f1be701209b5cf99cae33aec

File: ./contracts/VeMetis.sol
SHA3: 12591b1f7844aa73a850267aed84fde59b0405c2cb2fb2a549c286c69892eb95

Third review scope

Repository:
https://github.com/metisdac/Metis-DAC-contracts

Commit:
ae3c7b017f3469b3affdcc92ad9fbec7c8ca7ce7

Technical Documentation:
Type: Whitepaper (partial functional requirements provided)
Link

Integration and Unit Tests: Yes
Deployed Contracts Addresses: No
Contracts:

File: ./contracts/AddressList.sol
SHA3: 8b40e4fae1b94c31f5d6398ff89fed1dcb7b7b711ce77123545b35110a5a808f

File: ./contracts/ConfigKeys.sol
SHA3: 5475e2cee1eee905a6e37f7dc377eb3b039925b418cc93876799cbbe8301b7f3

File: ./contracts/Configuration.sol
SHA3: 0ffd25e8221c7ceca879c5f774ee634c36f5bf6b9b476a1500f68c1e50473dea

File: ./contracts/constants.sol
SHA3: 3aa139212193e48f7fa6e3b73edf841bbd9f43278c2e7a535d6ea51fd3341cc4

File: ./contracts/Controller.sol
SHA3: 6889b8f37d01ad5ade3a603bed06f1bced03e216329394cdc20b19696ee4849c

File: ./contracts/DAC.sol
SHA3: 72ca4261ad7745c85247f7fce348484dc817f8127760902116804e66e8f8c82f

File: ./contracts/DACFactory.sol
SHA3: 3f07f5e96723a62772a271e49ae8b56f9a2d3b2b101f943b671d725758f9a5d6

File: ./contracts/DACHelper.sol
SHA3: 81a442c975577756bc86f5535a8978347f605e5a5b939ac19ef75441ed9a6c88

File: ./contracts/EcoNode.sol
SHA3: d7627df970c35db18a8b6a712d31214e9fa6ff0b3a6c94ca0647061e9a8e5e14

File: ./contracts/ErrorCodes.sol
SHA3: 3cebbde3261b9c8cd53f5d4628106f02566b90760188d374fd87b4c00b2396de

File: ./contracts/interfaces/IBadgeManager.sol
SHA3: f7274fa5ea1c538402bd40be09a1b67c0be58182f22590f1c79b90d51882d1fb

```
File: ./contracts/interfaces/IController.sol
SHA3: c881a61eb0268357a7ad46c378b3be1715c38bca514e77d39323bce0d6438f81

File: ./contracts/interfaces/IDAC.sol
SHA3: 4b542b4633e804aabb5a9e0684f8b7a88c942a9cbfb0b18a67ead04fad34040e

File: ./contracts/interfaces/IDACFactory.sol
SHA3: 32153c5acfbcf7a7dc862a60d64eb8ce0dc0ce7ade82e8f0736d000a786771b0

File: ./contracts/interfaces/IDACHelper.sol
SHA3: 0abd27505740eb269eb78990a7ad1bf669d1d12c393ac0ad8138c5e22d957f2e

File: ./contracts/interfaces/IEcoNode.sol
SHA3: 83833321a5f5e267de3830efc35d9b68c8eb199d54a098d0d2f975b9fe97f59a

File: ./contracts/interfaces/IMetisNFTBadge.sol
SHA3: 45cbdbc2a3d8e9b7c924fb5f8a1bf134acb63eedc5a44190337f0bbc6df744c0

File: ./contracts/interfaces/IMetisNFTBadgeMetadata.sol
SHA3: 8bcc1a3e618477b7fd453dbc5ff7064f5b4d422107257972017dbabb79c51aa4

File: ./contracts/interfaces/IProposalFactory.sol
SHA3: fcf906c3558d3f86013cc1a3ef66c29caa5fbc123a8f51dbda976718119cf2b5

File: ./contracts/interfaces/IProxyFactory.sol
SHA3: 68d72c14b86104f956f1f2064df11c0986129ce8791e6a954e0943b6b8e47b40

File: ./contracts/interfaces/IRP.sol
SHA3: 2c39e024cba2ca9b54c4729b0ea7499ee38888aeff81cd1aa74eca095f673d4d

File: ./contracts/interfaces/ISimpleProposal.sol
SHA3: 4ead7576e32329c80a7b17ddd4fc83e45207f4d0ec531a614eb8f2bcabbf6093

File: ./contracts/interfaces/ISingleton.sol
SHA3: ed5aec341853cea7e3d3a717518a1a08b470095bd6b27d387051a4aa744f6e77

File: ./contracts/interfaces/ISingletonManager.sol
SHA3: 33a42fbff497f4e0bd7d2c8a22f026d1ff05af56850b73388c1a2cee4e08cd85

File: ./contracts/interfaces/safe.sol
SHA3: 763fc81b1f50be61539ab9c9f5775937af92f3c0035fb812efaafa18ab703be5

File: ./contracts/MasterCopyNames.sol
SHA3: 3165c5146208d349a295fffb4a86472fd803c3aabecb4cd542cb672cea4a98bc

File: ./contracts/MemberManager.sol
SHA3: e02280701dd7d738e634e7ae31872efef93553de1311acef4756f55cc3df5160

File: ./contracts/ProposalFactory.sol
SHA3: 7239a22a3485e53a0bbc305483e85ac0325759c495a570257cca624bd382dfb3

File: ./contracts/Proxy.sol
SHA3: a06380983916e426a4eb04f0ad905cf0c8fb5b5c616c98c2f316913b39b14a9e

File: ./contracts/ProxyFactory.sol
SHA3: 5201ab81128ed17b0ef83f043f6bb1f48c8b55fbd6d36a9fe6a4cac76022010b

File: ./contracts/Reward.sol
SHA3: ce8800a86d6a4edacf50e42293a3b3e6470cbc3b8c4f5e8da6130743d3871ba9
```

File: ./contracts/RP.sol
SHA3: 736bdc92c7e5c19f90ff1ce8ecbb1578e20dfa3f737f7500aefc9b6defdf28fc

File: ./contracts/SimpleProposal.sol
SHA3: c73c08a46586994ff013eecced2539dad9824acdeaf83a275accdef4ab68e607

File: ./contracts/Singleton.sol
SHA3: 9946a7a769fb8246f8843e53008c9e0a5e9559525fe27dbb8bcdad2a3ee16c7f

File: ./contracts/SingletonManager.sol
SHA3: b8c24035d760b48a76b6a9444d4fa4a36dd8c4e446c2f59d1c09bba0474ad463

File: ./contracts/test/MockBadge.sol
SHA3: bc3ee466d58d23c6a7bab9dcdebe392583d169ee80424ba0f2f0fe09c8ff5803

File: ./contracts/test/MockBadgeManager.sol
SHA3: ebc3bd1fb7f8e5e3275a769a1c984f691f2bc0479a449172ead521f30526f959

File: ./contracts/test/MockBadSafe.sol
SHA3: cc042d0eaac4ae806cc0acd97f62a7343be335bdd148675f2fe36e040080739c

File: ./contracts/test/MockToken.sol
SHA3: a4644672430ea7810aa2d3a4fd290a0eb59f17ee599e8cd8f83cf5c0c0cf86a4

File: ./contracts/util.sol
SHA3: c173ed9a7208724a07b560ad9309d52db98e7b90f1be701209b5cf99cae33aec

File: ./contracts/VeMetis.sol
SHA3: 12591b1f7844aa73a850267aed84fde59b0405c2cb2fb2a549c286c69892eb95

Fourth review scope

Repository:
https://github.com/metisdac/Metis-DAC-contracts
Commit:
37d1fbd19c87f0b197fe774761a1ea42bd345d3b
Technical Documentation:
Type: Whitepaper (partial functional requirements provided)
Link

Integration and Unit Tests: Yes
Deployed Contracts Addresses: No
Contracts:
https://github.com/hknio/Metis-DAC-contracts-478814710/tree/37d1fbd19c87f0b197fe774761a1ea42bd345d3b

File: ./contracts/AddressList.sol
SHA3: 3162544d50cd0ef6b6159434ab7816b53a28d7f820cadac5357291ea04507350

File: ./contracts/ConfigKeys.sol
SHA3: a823c07d1e177a8f9a32566bd45c0fcd1ecaff4e91ebda9c55397a8537464e6b

File: ./contracts/Configuration.sol
SHA3: 857b398a737f692fd03ce64b22bc6e3813df4632243cdecbb0b2a2cb186515e1

File: ./contracts/constants.sol
SHA3: 03f39623f41a7fe33e423e37b074b0e42155e400c7216f8e38e731b0a23e9370

File: ./contracts/Controller.sol
SHA3: 753091045b3333b9e54b75ad57ac9b7af5321a69fcfb1391bd66ef89416b50fa

```
File: ./contracts/DAC.sol
SHA3: 4990ef43d295498c646171c7b4bb0f22f8f16423c5119416de146141eeca8889

File: ./contracts/DACFactory.sol
SHA3: c348d5a52858bf83c6962522c27304063162d6d85ec2b9dcf192bae038544e38

File: ./contracts/DACHelper.sol
SHA3: 5539ac8f073a12641f6ea07f91acc135c7d8ce60dedf1c1ce665f7fda7e06edd

File: ./contracts/EcoNode.sol
SHA3: 87d3552cb2786ed8d25c9f74d46e07b656e8dc81f4414cfbf671159570d4c0f9

File: ./contracts/ErrorCodes.sol
SHA3: f4caa55dc5b33f7f3e093d90a315059082d70eee58586e05bf4e628b1bf9cfcf

File: ./contracts/interfaces/IBadgeManager.sol
SHA3: 0de6092891b0c4516c7667241259f3591decbb2ae2ebb6ff6d8577ae6aa9fabf

File: ./contracts/interfaces/IController.sol
SHA3: 4b4be8abb425b0a4c56fb385651b242da95f6c174e53afc93fdabd655103bfd0

File: ./contracts/interfaces/IDAC.sol
SHA3: 7e135265f7c0beb571223348fabaca0ee139722c2095ce71e37622fc8d12e38a

File: ./contracts/interfaces/IDACFactory.sol
SHA3: 5344c18610c597f9731ed79baa434d6cd68a95a05bff3e0b57d4a9d6674beba7

File: ./contracts/interfaces/IDACHelper.sol
SHA3: 620c40ea3d21ee218a17c3ff50d15def446826fad51f6205f77b57bf1bdeb100

File: ./contracts/interfaces/IEcoNode.sol
SHA3: 257fb8dc2be00c071584e6064a58e09ec0aade3dba9147b21e86413fe7511b5b

File: ./contracts/interfaces/IMetisNFTBadge.sol
SHA3: 906edb62924aa9d226fd7a4c343cd192d142bd5ea8e8d8feb545c2f5f87b4305

File: ./contracts/interfaces/IMetisNFTBadgeMetadata.sol
SHA3: 9b5d8a5921bdf3aa0d98be457a049db446a73d558db3af5ba83018f7ab1eb517

File: ./contracts/interfaces/IProposalFactory.sol
SHA3: ca647c381f215b63d374795b7c8daf456509a145020f65e82151b31536bd0717

File: ./contracts/interfaces/IProxyFactory.sol
SHA3: 463f724d4273d78d7c415eddf9c5158d065303907a4df65d5b099af8edf49081

File: ./contracts/interfaces/IRP.sol
SHA3: 2b7cb2b8ef70b7e554d430e84b51037fdb97b0b86a8b81b038a9f2d8b02d8474

File: ./contracts/interfaces/ISimpleProposal.sol
SHA3: 40867630942df0fbcda1a6655937ebd393db8e64405b03fd121773fc2e4873e9

File: ./contracts/interfaces/ISingleton.sol
SHA3: 3ff335c86be4a0a5dc69cedeb7b0d7b9e8b8beba4ec281ea074e55cb6252499b

File: ./contracts/interfaces/ISingletonManager.sol
SHA3: ed73d0fe96ceb49e48e4db507f2bef7736b96f7c1b15d29194d966a129c0e2e1

File: ./contracts/interfaces/safe.sol
SHA3: 86689d54d4cdf8cdb6221f97e9e00576cfe6578dabc4e833f9e890594e02df9d
```

File: ./contracts/MasterCopyNames.sol
SHA3: 4b49ba06b26ceba64935a708c4c51b19ec6a8c7b58b443452f62149165a52a87

File: ./contracts/MemberManager.sol
SHA3: cdff655f73877262523afd896a443ebdb867adb86b3a0b7bca28655b0f1914fb

File: ./contracts/ProposalFactory.sol
SHA3: 51ea03dcf0f55f8381a845923cc6243829942902cb7ccd83983fe642eb87e6bb

File: ./contracts/Proxy.sol
SHA3: 82db2790af38adb8cf15df7d2f4fe2f457129fbc836800d18dff6e8a556a058a

File: ./contracts/ProxyFactory.sol
SHA3: a3c389521ac9ce5fe7295d109a0e15bb213352962749d76d4446f25b860c7103

File: ./contracts/Reward.sol
SHA3: 18da0819b0799a14427b15fdd2d6c136d3cd8b766d0d3bb875e29af00d63769a

File: ./contracts/RP.sol
SHA3: dce2d378be1098760675833a6b3004b8ab4e4d941f9c4e5a78091ede6c75d459

File: ./contracts/SimpleProposal.sol
SHA3: 6b92668942fefc805652761960cf50df96a6c62d9bb73c74b43b221d641b8406

File: ./contracts/Singleton.sol
SHA3: fc7720ca08aa129b1a3dbeeb1e9eb10a2d72986568025a8289f00eefe0b9d528

File: ./contracts/SingletonManager.sol
SHA3: 13f35eb39ea0fbac8d72715c13324f1d544cf78effc86c28b600d5ba5ce2f09a

File: ./contracts/test/MockBadge.sol
SHA3: b17cc3785c312d264802df1f27905df27e6ab51dc4c8b20885abbbea3893f329

File: ./contracts/test/MockBadgeManager.sol
SHA3: 3ab91df830ad61d1d4d00d2440a9d6c36e157524ffea50094c9c6d581ebb94be

File: ./contracts/test/MockBadSafe.sol
SHA3: c527933625b2d1e05a433aaddf9f1ae9060c26d096b36c115de02e0895b5eb27

File: ./contracts/test/MockToken.sol
SHA3: 4ffb3aed9e90723e27bacc4d7664a2ee333d85c77cb1509fb6c48acb009add96

File: ./contracts/util.sol
SHA3: f955860179234df042c66dbe6984d2f0109c92b9872276074c1c8f5c901ea435

File: ./contracts/VeMetis.sol
SHA3: 8ac4d409214fe7c7eb4a9b634e1afa15a08dd2d9d85e9abb33d137c91e01566f

Fifth review scope

Repository:
https://github.com/metisdac/Metis-DAC-contracts

Commit:
678222c842704ccddf9f29db7a7330f8f127b490

Technical Documentation:
Type: Whitepaper (partial functional requirements provided)
Link

Integration and Unit Tests: Yes
Deployed Contracts Addresses: No

Contracts:

File: ./contracts/AddressList.sol
SHA3: d8275d519e16813fe7d82e7c0159a674818af697bf5e0dd6a57890020882826f

File: ./contracts/ConfigKeys.sol
SHA3: a5470f349da0e0b559400fd966836af3199c2d02a05a8e9eba46fcf834cb62d3

File: ./contracts/Configuration.sol
SHA3: b4aa93beaaec83ba2680848a5c657b7841562273864d349c97c1156d4cdf0ed9

File: ./contracts/constants.sol
SHA3: 03f39623f41a7fe33e423e37b074b0e42155e400c7216f8e38e731b0a23e9370

File: ./contracts/Controller.sol
SHA3: 76474117a75749cf651ee9264356259bafda9ee71f3bf6f8e75cbf98f12ee47c

File: ./contracts/DAC.sol
SHA3: 256e4a01eb57dce0b05cc27095fc75bf663839a8dfbbc6eae555a33aba735775

File: ./contracts/DACFactory.sol
SHA3: 1c7f6ece32beb5adb52497b27e3106ac2946f706e6c23e5c5aecae7c20363756

File: ./contracts/DACHelper.sol
SHA3: 7e8de6b696522ccba8d86b6ed0c136ea9e7bd61467be866c205f33626e32b811

File: ./contracts/EcoNode.sol
SHA3: b76ae30e431d60be1886789935f425208dbcb56ab9fca824d250eea0e59141a0

File: ./contracts/ErrorCodes.sol
SHA3: f4caa55dc5b33f7f3e093d90a315059082d70eee58586e05bf4e628b1bf9cfcf

File: ./contracts/interfaces/IBadgeManager.sol
SHA3: 0de6092891b0c4516c7667241259f3591decbb2ae2ebb6ff6d8577ae6aa9fabf

File: ./contracts/interfaces/IController.sol
SHA3: 4b4be8abb425b0a4c56fb385651b242da95f6c174e53afc93fdabd655103bfd0

File: ./contracts/interfaces/IDAC.sol
SHA3: 95465a7b51ecd7afda0dd95b0854971847d952dad5bca602d436143ba1ac158d

File: ./contracts/interfaces/IDACFactory.sol
SHA3: ffe908887ff7443d8546c9a01dbb98f4d162d78083d4376751d093e6e00f9e39

File: ./contracts/interfaces/IDACHelper.sol
SHA3: 7e2fddd4b9f914873dde6ce22a1e107f6bf17d11a6748a802a6242e9cd3d3275

File: ./contracts/interfaces/IEcoNode.sol
SHA3: 66c8b216e9ad582234c4966474b8c316d90b42c2a017603fc8c00cef99eaf6b9

File: ./contracts/interfaces/IMetisNFTBadge.sol
SHA3: 70b3a48e1e1c91c892b61df3afd31605250dfe64f0935721b331e034f6a22779

File: ./contracts/interfaces/IMetisNFTBadgeMetadata.sol
SHA3: aad136b1082964fd2ad00a68ee2e1aef7b9492b95d1e980dca1ca54bf8cfa98f

File: ./contracts/interfaces/IProposalFactory.sol
SHA3: dde725add9c35d3b471fc52b4d8908f4ea95792672ac864be422fdbfa2babe35

File: ./contracts/interfaces/IProxyFactory.sol
SHA3: 463f724d4273d78d7c415eddf9c5158d065303907a4df65d5b099af8edf49081

```
File: ./contracts/interfaces/IRP.sol
SHA3: 380e96c6ef878ded61b9391bb224b73e4f40849519d86464ec3d664227f7ed66

File: ./contracts/interfaces/ISimpleProposal.sol
SHA3: c4e3e72fa99f94feab15734bbc749360504c899503737509d383642315c24b99

File: ./contracts/interfaces/ISingleton.sol
SHA3: 3ff335c86be4a0a5dc69cedeb7b0d7b9e8b8beba4ec281ea074e55cb6252499b

File: ./contracts/interfaces/ISingletonManager.sol
SHA3: ed73d0fe96ceb49e48e4db507f2bef7736b96f7c1b15d29194d966a129c0e2e1

File: ./contracts/interfaces/safe.sol
SHA3: 86689d54d4cdf8cdb6221f97e9e00576cfe6578dabc4e833f9e890594e02df9d

File: ./contracts/MasterCopyNames.sol
SHA3: 4b49ba06b26ceba64935a708c4c51b19ec6a8c7b58b443452f62149165a52a87

File: ./contracts/MemberManager.sol
SHA3: d9b3fb103c202838027602ff5501aa6a353203b91257995c910907508d3c61c8

File: ./contracts/ProposalFactory.sol
SHA3: a658201fc1677a03a096bc5c618c82525ef73e626bd5949e0c56205d54ab977a

File: ./contracts/Proxy.sol
SHA3: bcb3453aa427da859ef8954139ae1cb7f7f0727ac30b231f17e3107ff738f902

File: ./contracts/ProxyFactory.sol
SHA3: a3c389521ac9ce5fe7295d109a0e15bb213352962749d76d4446f25b860c7103

File: ./contracts/Reward.sol
SHA3: 562bf7ad7c58416eb5587563c759b2b8e7ce0b8cdeb924d4fac1cb912987e412

File: ./contracts/RP.sol
SHA3: ef093372efd82c7f8faeb27f7a2e74684f2d7ce3e6adb9769bfed507bd555d10

File: ./contracts/SimpleProposal.sol
SHA3: 66fa669c2f80d0c19a5ba902e48005f19a81a1395a7f34867ca9235ddd4749b8

File: ./contracts/Singleton.sol
SHA3: 6e0298d41e2c13dddce97e63fbae58f909e6e2473b1520e2c69f5ab5bad46216

File: ./contracts/SingletonManager.sol
SHA3: 13f35eb39ea0fbac8d72715c13324f1d544cf78effc86c28b600d5ba5ce2f09a

File: ./contracts/test/MockBadge.sol
SHA3: b17cc3785c312d264802df1f27905df27e6ab51dc4c8b20885abbbea3893f329

File: ./contracts/test/MockBadgeManager.sol
SHA3: 3ab91df830ad61d1d4d00d2440a9d6c36e157524ffea50094c9c6d581ebb94be

File: ./contracts/test/MockBadSafe.sol
SHA3: c527933625b2d1e05a433aaddf9f1ae9060c26d096b36c115de02e0895b5eb27

File: ./contracts/test/MockToken.sol
SHA3: 4ffb3aed9e90723e27bacc4d7664a2ee333d85c77cb1509fb6c48acb009add96

File: ./contracts/util.sol
SHA3: 9f1f65c04f23e97a63192acc57b0add64f70ef117a5aceb8593f1aaef5fa6cff
```

```
File: ./contracts/VeMetis.sol
SHA3: 8ac4d409214fe7c7eb4a9b634e1afa15a08dd2d9d85e9abb33d137c91e01566f
```

Severity Definitions

Risk Level	Description
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to assets loss or data manipulations.
High	High-level vulnerabilities are difficult to exploit; however, they also have a significant impact on smart contract execution, e.g., public access to crucial functions.
Medium	Medium-level vulnerabilities are important to fix; however, they cannot lead to assets loss or data manipulations.
Low	Low-level vulnerabilities are mostly related to outdated, unused, etc. code snippets that cannot have a significant impact on execution.

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 methodology.

Documentation quality

The total Documentation Quality score is **7** out of **10**. Functional requirements are partially missed. A technical description that demonstrates the function explanations was not provided. A public whitepaper is provided.

Code quality

The total CodeQuality score is **7** out of **10**. Unit tests were provided, but some of them failed. Style guide violation is detected. NatSpec format was mostly followed. **Test coverage is 86.87%**.

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

Architecture quality

The architecture quality score is **7** out of **10**. Hardhat is used as a development environment and deployment was successful. Instructions were provided in readme. Some unused template file was detected in the project.

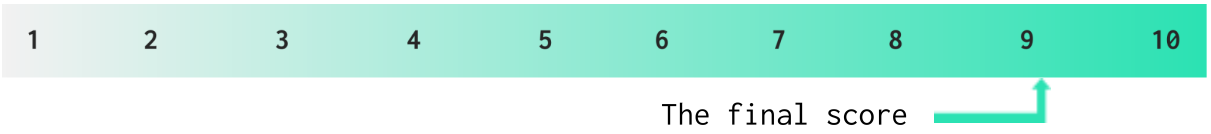
Security score

As a result of the audit, the code contains **1** low severity issue. 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: **9.1**.



Checked Items

We have audited provided smart contracts for commonly known and more specific vulnerabilities. Here are some of the items that are 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.	Passed
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.	Passed
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.	Passed
DoS (Denial of Service)	SWC-113 SWC-128	Execution of the code should never be blocked by a specific contract state unless it is required.	Passed
Race Conditions	SWC-114	Race Conditions and Transactions Order Dependency should not be possible.	Passed
Authorization	SWC-115	tx.origin should not be used for	Passed

through tx.origin		authorization.	
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	Signed messages should always have a unique id. A transaction hash should not be used as a unique id. Chain identifier should always be used. All parameters from the signature should be used in signer recovery	Passed
Shadowing State Variable	SWC-119	State variables should not be shadowed.	Passed
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.	Passed
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.	Passed
Assets integrity	**Custom**	Funds are protected and cannot be withdrawn without proper permissions.	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.	Not Relevant
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	Passed

		data stored on the contract. There should not be any cases when execution fails due to the block Gas limit.	
Style guide violation	**Custom**	Style guides and best practices should be followed.	Failed
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.	Not Relevant
Tests Coverage	**Custom**	The code should be covered with unit tests. Test coverage should be 100%, with both negative and positive cases covered. Usage of contracts by multiple users should be tested.	Failed
Stable Imports	**Custom**	The code should not reference draft contracts, that may be changed in the future.	Passed

System Overview

DAC Portal is a Decentralized Autonomous Company(DAO/DAC) that allows users to create their DACs or join an existing DAC. The project has different feature than the EVM compatible networks; the project's main tokens, Metis ERC20 and native token act simultaneously and they can be considered as same tokens. The system consists of the following contracts:

- *AddressList* – a library contract that helps store and manage addresses of DAC members and owners.
- *Configuration* – a simple contract that sets the gnosisSingleton, gnosisSafeProxyFactory, ecoNodeToken, endorseCoolDown addresses by the owner.
- *constants* – a Solidity file that keeps all the constant values for the project.
- *Controller* – a simple contract that sets the controller proxy(such as DAC Factory or Member Manager) keys with values as their addresses.
- *DAC* – a contract that is used to create a DAC(Decentralized Autonomous Company or used by users to join an existing DAC or to leave from a subscripted DAO.
- DACFactory – an upgradable contract that generates a new deployed DAC contract for each user with the given admin addresses or allows the DAC owner to dismiss its DAC contract and sends the user's balance to the admin's address.
- DACHelper – a contract that helps to view DAC properties, members, membership fee, staked users, and the amounts.
- EcoNode – an upgradable contract that allows users to stake Metis or native tokens and allows them to unstake tokens and claim rewards.
- MemberManager – a contract that manages members of DACs while adding or removing, and creates invitation schema.
- Proxy – a contract that helps to deal with the storages for upgradable contracts.
- ProxyFactory – a contract that generates(deploys) a new ControllerProxy within the given lookup name.
- ProposalFactory – a factory contract that generates a new deployed SimpleProposal contract
- Reward – an ownable contract that manages staking, unstaking and allows the owner to set the reward amount and the duration.
- RP – an upgradable contract that allows users to endorse DACs with NFTs.
- SimpleProposal – a contract that allows DAC members to vote for a proposal in direct proportion to their contributions.
- Singleton – a simple abstract contract that sets the Singleton Manager address.
- SingletonManager – an upgradable contract that manages proxy factory and the singletons.

- util – a library contract that converts uppercase characters into lowercase characters to prevent users from having the same DAC names.
- VeMetis – a burnable and mintable ERC20 token contract that cannot be transferred by users.
  It has the following properties:
  Name: veMetis
  Symbol: veMetis

Privileged roles

- The admin of the *DAC* contract can:
  - specify DAC's managers and properties
  - set a membership fee, fee type, and Badge membership address.
- The owner of the EcoNode contract can:
  - change the reward amount
  - change the reward duration
  - withdraw any ERC20 token balance of the contract
- The owner of the Configuration contract can:
  - set the Gnosis Singleton address
  - set the Gnosis Safe Proxy address
  - set the EcoNode token address
  - set the EndorseCoolDown address
- The owner of the DACFactory contract can:
  - update the contract

Risks

- The given contracts can work properly only on Metis network and cannot be used on EVM or EVM compatible networks since it has a different working mechanism. In Metis network, ERC20 tokens and native tokens of Metis can be considered the same. Therefore, when transferring any amount of ERC20 tokens to an address, the network sends the same amount of native tokens to the same address and vice versa.

Findings

■■■■ Critical

1. Incorrect Function Logic

err is assigned as a result of the *checkDismiss* function execution, and the value is always equal to the *DISERR_UNKNOWN* constant. However, to call the dismiss function, *err* must be equal to the *DISERR_SUCCESS* constant.

dismiss function could not be executed.

File: ./contracts/DAC.sol

Contract: DAC

Function: dismiss

Recommendation: Replace the require statement with a meaningful condition.

Status: Fixed (Revised commit: cab2df387ea7d9b8219e7056890b1bf967a7e3ae)

2. Insufficient Balance

Whether Metis token is staked or native token is staked, the amounts are recorded in the same mapping by making an external call to the Reward contract (Lines 64-65). The *unstake* function withdraws only Metis token, regardless of what a user actually staked.

The possibility to withdraw native tokens is missing.

File: ./contracts/EcoNode.sol

Contract: EcoNode

Function: unstake

Recommendation: Treat native coin and token transfers differently.

Status: Mitigated (with Customer notice) (Revised commit: ae3c7b017f3469b3affdcc92ad9fbec7c8ca7ce7)

3. Unauthorized Access

The function does not have proper authorization and can be called by anyone.

The contract state can become inconsistent.

File: ./contracts/MasterDeployer.sol

Contract: MasterDeployer

Function: upgrade

Recommendation: Implement authorization control functionality.

Status: Fixed (Revised commit: cab2df387ea7d9b8219e7056890b1bf967a7e3ae)

4. **Denial of Service**

`_beforeTokenTransfer` function has a condition on lines 40-43 that blocks token transfers.

Making `from address` or `to address` zero address means only burning and minting are allowed.

File: ./contracts/VeMetis.sol

Contract: VeMetis

Function: _beforeTokenTransfer

Recommendation: Remove the *require* statement.

Status: Mitigated (with Customer notice) (Revised commit: cab2df387ea7d9b8219e7056890b1bf967a7e3ae)

High

1. **Insufficient Balance**

When a reward is claimed, the `_claim` function distributes Metis tokens without checking if the contract has more balance than the total staked tokens.

If the contract's Metis token balance becomes less than the total staked amount of the users, the staked funds can be lost.

File: ./contracts/EcoNode.sol

Contract: EcoNode

Function: _claim

Recommendation: Check the contract balance before sending the rewards.

Status: Fixed (Revised commit: ae3c7b017f3469b3affdcc92ad9fbec7c8ca7ce7)

2. **Highly Permissive Role Access**

In `recoverERC20` function, every token can be withdrawn by the owner, including the stake token.

This may lead users to lose their staked tokens.

File: ./contracts/EcoNode.sol

Contract: EcoNode

Function: recoverERC20

Recommendation: During executing recoverERC20, check if the token is Metis. For this case, allow withdrawing only excess from staking tokens.

Status: Fixed (Revised commit: cab2df387ea7d9b8219e7056890b1bf967a7e3ae)

3. Denial of Service Vulnerability

updateEndorsements function always queries endorsement list between 0 and 999 index. Moreover, it is impossible to view beyond the 999th index of the list.

Large iterations may cause failed transaction due to exceeding Gas.

File: ./contracts/RP.sol

Contract: RP

Function: updateEndorsements

Recommendation: Make the start address and the limit mutable instead of giving default values.

Status: Fixed (Revised commit: cab2df387ea7d9b8219e7056890b1bf967a7e3ae)

4. Denial of Service Vulnerability

memberList functions always start from the zero index to query list elements.

Therefore, when the array is large enough, it will be impossible to view it due to exceeding Gas.

File: ./contracts/MemberManager.sol

Contract: MemberManager

Function: memberList

Recommendation: Make the start address mutable and allow querying in batches.

Status: Fixed (Revised commit: cab2df387ea7d9b8219e7056890b1bf967a7e3ae)

5. Data Consistency

There is no check for the token addresses parameter in *endorse_* function. Users can endorse with any NFT address they want. Moreover, they can deploy the same contract and endorse unlimitedly.

This may lead to data inconsistency and manipulation.

File: ./contracts/RP.sol

Contract: RP

Function: endorse_

Recommendation: Validate the token address.

Status: Fixed (Revised commit: cab2df387ea7d9b8219e7056890b1bf967a7e3ae)

6. **Funds Lock**

   Contract DAC and EcoNode have payable functions but do not have a function to withdraw the Ether.

   Files: ./contracts/DAC.sol, ./contracts/EcoNode.sol

   Contracts: DAC, EcoNode

   Functions: -

   Recommendation: Write functions to withdraw the locked Ether.

   Status: Mitigated (with Customer notice) (Revised commit: ae3c7b017f3469b3affdcc92ad9fbec7c8ca7ce7)

7. **Highly Permissive Role Access**

   The existing implementation of the VeMetis token allows the owner to burn tokens from any address. This behavior violates the rule that owner should not have access to funds that belongs to users.

   File: ./contracts/VeMetis.sol

   Contract: VeMetis

   Function: burnFrom

   Recommendation: Check allowance before tokens burning.

   Status: Mitigated (with Customer notice) (Revised commit: cab2df387ea7d9b8219e7056890b1bf967a7e3ae)

Medium

1. **Data Consistency**

   After dismissing a DAC, the IDs and the names in *dacById* and *dacNames* mappings are not updated as mapping to zero address.

   Therefore, IDs and names will always be occupied, although their DACs are destroyed.

   File: ./contracts/DACFactory.sol

   Contract: DACFactory

   Function: dismissDAC

   Recommendation: Update the related mappings after a dismiss.

   Status: Fixed (Revised commit: cab2df387ea7d9b8219e7056890b1bf967a7e3ae)

2. **Violated Checks-Effects-Interactions Pattern**

State is updated on lines 64-65 before the stake amount is received.

File: ./contracts/EcoNode.sol

Contract: EcoNode

Function: _stake

Recommendation: Ensure all internal state changes are performed before an external call execution.

Status: Fixed (Revised commit: cab2df387ea7d9b8219e7056890b1bf967a7e3ae)

3. **Unchecked Transfer**

The return value of an external *transfer*/*transferFrom* call is not checked. Several tokens do not revert in case of failure and return false. If one of these tokens is used in audited contracts, the withdrawal will not revert if the transfer fails, and an attacker can call the withdrawal for free.

File: ./contracts/EcoNode.sol

Contract: EcoNode

Functions: _stake, unstake, _claim,

Recommendation: Check the result of the transfer if it is true or not.

Status: Fixed (Revised commit: cab2df387ea7d9b8219e7056890b1bf967a7e3ae)

4. **Overcomplicated Implementation**

The function fetches all owners and iterates over this list.

This process can be simplified, and Gas consumption will be decreased.

File: ./contracts/DAC.sol

Contract: DAC

Function: isOneOfAdmins

Recommendation: Use *isOwner* function of the GnosisSafe contract

Status: Fixed (Revised commit: cab2df387ea7d9b8219e7056890b1bf967a7e3ae)

5. **Redundant Import**

hardhat/console.sol must not be in the deployment version of the contract.

Contract: veSeaNFTV2.sol

Function: -

Recommendation: Remove *console.sol* imports and usages before the deployment.

Status: Fixed (Revised commit: cab2df387ea7d9b8219e7056890b1bf967a7e3ae)

6. Incorrect Function Logic

stake function allows staking with Metis token, and *stakePay* function does it with the native token.
Require statement on line 56 should have been used in the *stake* function because it is the one that manages the Metis tokens for staking.

File: ./contracts/EcoNode.sol

Contract: EcoNode

Functions: stake, stakePay

Recommendation: Move the require statement in the *stake* function.

Status: Mitigated (with Customer notice) (Revised commit: ae3c7b017f3469b3affdcc92ad9fbec7c8ca7ce7)

Low

1. Redundant Code Line

Although there is already a function implementation to check if an address exists, the same code implementation is re-written on line 15.

Using the existing implementation increases the code readability.

File: ./contracts/AddressList.sol

Contract: AddressList

Function: add

Recommendation: Use the internal *exists* function instead of re-writing the same lines.

Status: Fixed (Revised commit: cab2df387ea7d9b8219e7056890b1bf967a7e3ae)

2. Public Functions Instead of External

Some of the functions are declared as public, although they are not called internally in the related contract.

Public function visibility consumes more Gas than external visibility.

Files: ./contracts/Controller.sol, ./contracts/DAC.sol, ./contracts/DACFactory.sol

Contracts: Controller, DAC, DACFactory

Functions: Controller.setValue, DAC.logo, DAC.desc, DACFactory.createDAC

Recommendation: Change public visibility with external.

Status: Fixed (Revised commit: cab2df387ea7d9b8219e7056890b1bf967a7e3ae)

3. **Style Guide Violation**

Contracts do not follow the Solidity code style guide.

Files: all

Contracts: all

Recommendation: Follow the official Solidity code style guide.

Status: Reported (Revised commit: cab2df387ea7d9b8219e7056890b1bf967a7e3ae)

4. **Redundant Function Usage**

isMember and *joined* functions are doing the same thing.

Redundant usages cost more Gas and decrease the code readability.

File: ./contracts/DAC.sol

Contract: DAC

Functions: isMember, joined

Recommendation: Remove *joined* function.

Status: Fixed (Revised commit: cab2df387ea7d9b8219e7056890b1bf967a7e3ae)

5. **Duplicated Functions**

isOneOfAdmins function and *_isOneOfAdmins* function have exactly the same implementation.

Unnecessary code implementation costs more Gas and decreases the code readability.

File: ./contracts/DAC.sol

Contract: DAC

Functions: isOneOfAdmins, _isOneOfAdmins

Recommendation: Remove one of the functions.

Status: Fixed (Revised commit: cab2df387ea7d9b8219e7056890b1bf967a7e3ae)

6. **Confusing Function Name**

allDACs function can be understood as viewing all DAC addresses, which can confuse users.

File: ./contracts/DACFactory.sol

Contract: DACFactory

Function: allDACs

Recommendation: Change the function name with an appropriate one, such as *viewDAC/viewDACById*.

Status: Fixed (Revised commit: cab2df387ea7d9b8219e7056890b1bf967a7e3ae)

7. Unused Modifier

Despite the modifier *checkOrigin* being declared, it is not used anywhere.

Redundant code lines cost more Gas and decrease the code readability.

File: ./contracts/DACHelper.sol

Contract: DACHelper

Function: -

Recommendation: Remove the *checkOrigin* modifier.

Status: Fixed (Revised commit: cab2df387ea7d9b8219e7056890b1bf967a7e3ae)

8. Unused Variable

Local *logoKey* variable is declared but never used.

Redundant declarations cost more Gas and decrease the code readability.

File: ./contracts/DACHelper.sol

Contract: DACHelper

Function: userJoinedList

Recommendation: Remove the *logoKey* variable declaration.

Status: Fixed (Revised commit: cab2df387ea7d9b8219e7056890b1bf967a7e3ae)

9. Empty Functions

dacInfo and *stakedMembers* functions do not have any implementation.

Redundant declarations cost more Gas and decrease the code readability.

File: ./contracts/Reward.sol

Contract: Reward

Functions: dacInfo, stakedMembers

Recommendation: Remove the functions or write the necessary implementation.

Status: Fixed (Revised commit: cab2df387ea7d9b8219e7056890b1bf967a7e3ae)

10. Redundant Function

Internal *time* function is declared but never used anywhere.

File: ./contracts/util.sol

Contract: util

Function: time

Recommendation: Remove the unused function.

Status: Fixed (Revised commit: cab2df387ea7d9b8219e7056890b1bf967a7e3ae)

11. Redundant Use of SafeMath

Since Solidity v0.8.0, the overflow/underflow check is implemented via ABIEncoderV2 on the language level - it adds the validation to the bytecode during compilation.

There is no need to use the SafeMath library.

Files: ./contracts/EcoNode.sol, ./contracts/Reward.sol

Contracts: EcoNode, Reward

Function: -

Recommendation: Remove SafeMath.

Status: Fixed (Revised commit: cab2df387ea7d9b8219e7056890b1bf967a7e3ae)

12. 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.

Files: all

Contracts: all

Function: -

Recommendation: Consider locking the pragma version to the latest one and avoid using a floating pragma in the final deployment.

Status: Fixed (Revised commit: cab2df387ea7d9b8219e7056890b1bf967a7e3ae)

13. Missing Zero Address Validation

The specified functions do not check if the given parameter is a zero address, which could be sent as a default value.

Files: ./contracts/Configuration.sol, ./contracts/SingletonManager.sol, ./contracts/Singleton.sol, ./contracts/EcoNode.sol, ./contracts/RP.sol, ./contracts/DACFactory.sol

Contracts: Configuration, SingletonManager, EcoNode, RP, DACFactory

Functions: Configuration.setGnosisSingleton, Configuration.setGnosisSafeProxyFactory, Configuration.setEcoNodeToken, Configuration.setEndorseCoolDown, SingletonManager.initialize, SingletonManager.setProxyFactory, Singleton.initialize, EcoNode.initialize, RP.initialize, DACFactory.createDAC

Recommendation: Add a statement for the address parameters to check if it is zero address.

Status: Fixed (Revised commit: ae3c7b017f3469b3affdcc92ad9fbec7c8ca7ce7)

14. Redundant Hardcoded Value

In `_checkQuit` function, `true` boolean value is assigned to `result[0]` and it is hardcoded. So, this value will not affect anything when a member wants to quit.
Redundant code causes consumption of more Gas and decreases the code readability.

File: ./contracts/DAC.sol

Contract: DAC

Function: _checkQuit

Recommendation: Remove the redundant value.

Status: Reported (Revised commit: 1ec034fef5f8c3f40b844b5f0bed115a4c22730f)

15. Redundant Comparison

`CFG_MS_FEE_LIMIT` is already checked in `addContribution` function. Therefore, there is no need to check it again on line 185.
Redundant code causes consumption of more Gas and decreases the code readability.

File: ./contracts/DAC.sol

Contract: DAC

Function: _join

Recommendation: Delete the redundant comparison.

Status: Fixed (Revised commit: 678222c842704ccddf9f29db7a7330f8f127b490)

Disclaimers

Hacken Disclaimer

The smart contracts given for audit have been analyzed by the best industry practices at the date 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 to 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, Consultant cannot guarantee the explicit security of the audited smart contracts.