SOLIDProof

Bring trust into your projects

Blockchain Security | Smart Contract Audits | KYC

MADE IN GERMANY

HelixMeta

Audit

Security Assessment

30. April, 2022

For

Disclaimer

SolidProof.io reports are not, nor should be considered, an "endorsement" or "disapproval" of any particular project or team. These reports are not, nor should be considered, an indication of the economics or value of any "product" or "asset" created by any team. SolidProof.io do not cover testing or auditing the integration with external contract or services (such as Unicrypt, Uniswap, PancakeSwap etc'...)

SolidProof.io Audits do not provide any warranty or guarantee regarding the absolute bug- free nature of the technology analyzed, nor do they provide any indication of the technology proprietors. SolidProof Audits should not be used in any way to make decisions around investment or involvement with any particular project. These reports in no way provide investment advice, nor should be leveraged as investment advice of any sort.

SolidProof.io Reports represent an extensive auditing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology. Blockchain technology and cryptographic assets present a high level of ongoing risk. SolidProof's position is that each company and individual are responsible for their own due diligence and continuous security. SolidProof in no way claims any guarantee of security or functionality of the technology we agree to analyze.

Version	Date	Description
1.0	30. April 2022	• Layout project • Automated- /Manual-Security Testing • Summary

Network

Ethereum (ERC20)

Website

https://www.helixmeta.org/

Telegram

https://t.me/helixmeta

Twitter

https://mobile.twitter.com/helix_meta

Medium

https://helixmeta.medium.com/

Discord

https://discord.gg/9CR4Kpur6Y

Description

HelixMeta is the leading NFT marketplace with participating rewards.

Project Engagement

During the 19th of April 2022, **HelixMeta Team** engaged Solidproof.io to audit smart contracts that they created. The engagement was technical in nature and focused on identifying security flaws in the design and implementation of the contracts. They provided Solidproof.io with access to their code repository and whitepaper.

Logo

Contract Link

v1.0

- Github
  - https://github.com/HelixMeta/helixmeta-core
  - Commit: c7e91056ba2ed67689790cd7b1ce68d0780949cc

Vulnerability & Risk Level

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

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

Auditing Strategy and Techniques Applied

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

Methodology

The auditing process follows a routine series of steps:

1. Code review that includes the following:
    i) Review of the specifications, sources, and instructions provided to SolidProof to make sure we understand the size, scope, and functionality of the smart contract.
    ii) Manual review of code, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
    iii) Comparison to specification, which is the process of checking whether the code does what the specifications, sources, and instructions provided to SolidProof describe.

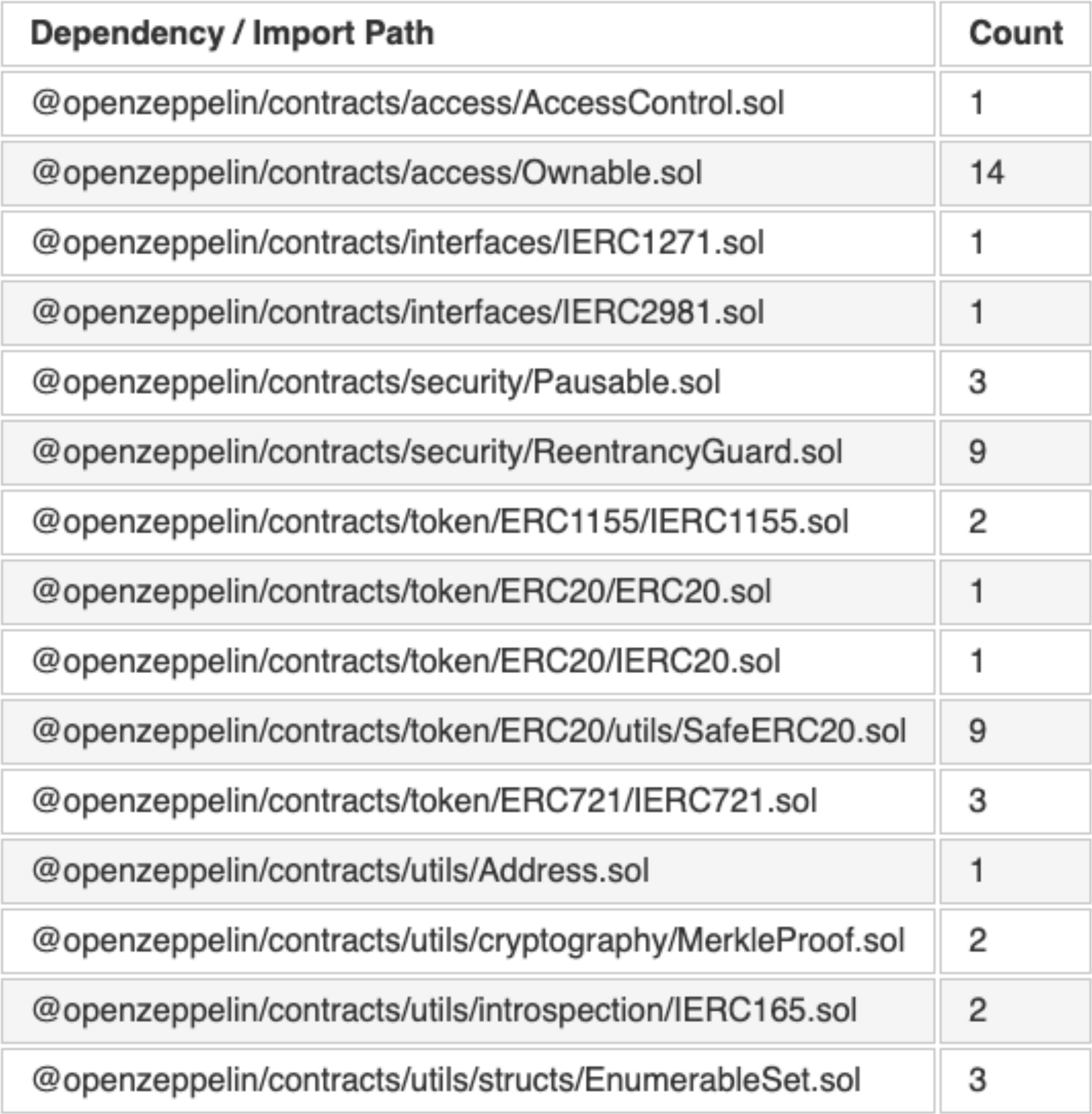
2. Testing and automated analysis that includes the following:
    i) Test coverage analysis, which is the process of determining whether the test cases are actually covering the code and how much code is exercised when we run those test cases.
    ii) Symbolic execution, which is analysing a program to determine what inputs causes each part of a program to execute.

3. Best practices review, which is a review of the smart contracts to improve efficiency, effectiveness, clarify, maintainability, security, and control based on the established industry and academic practices, recommendations, and research.

4. Specific, itemized, actionable recommendations to help you take steps to secure your smart contracts.

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

Imported packages:



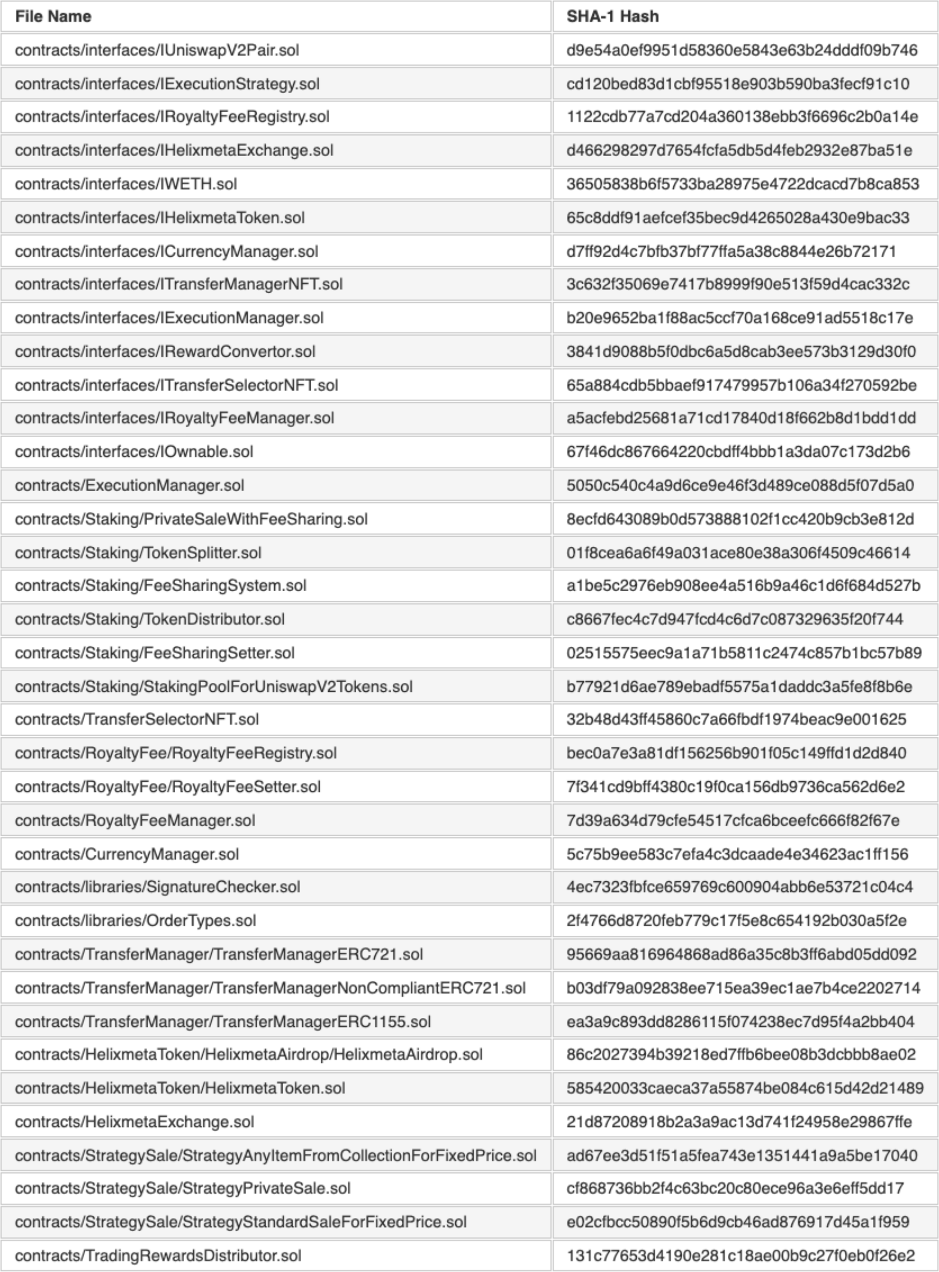
Dependency / Import Path	Count
@openzeppelin/contracts/access/AccessControl.sol	1
@openzeppelin/contracts/access/Ownable.sol	14
@openzeppelin/contracts/interfaces/IERC1271.sol	1
@openzeppelin/contracts/interfaces/IERC2981.sol	1
@openzeppelin/contracts/security/Pausable.sol	3
@openzeppelin/contracts/security/ReentrancyGuard.sol	9
@openzeppelin/contracts/token/ERC1155/IERC1155.sol	2
@openzeppelin/contracts/token/ERC20/ERC20.sol	1
@openzeppelin/contracts/token/ERC20/IERC20.sol	1
@openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol	9
@openzeppelin/contracts/token/ERC721/IERC721.sol	3
@openzeppelin/contracts/utils/Address.sol	1
@openzeppelin/contracts/utils/cryptography/MerkleProof.sol	2
@openzeppelin/contracts/utils/introspection/IERC165.sol	2
@openzeppelin/contracts/utils/structs/EnumerableSet.sol	3

Tested Contract Files

This audit covered the following files listed below with a SHA-1 Hash.

A file with a different Hash has been modified, intentionally or otherwise, after the security review. A different Hash could be (but not necessarily) an indication of a changed condition or potential vulnerability that was not within the scope of this review.

v1.0

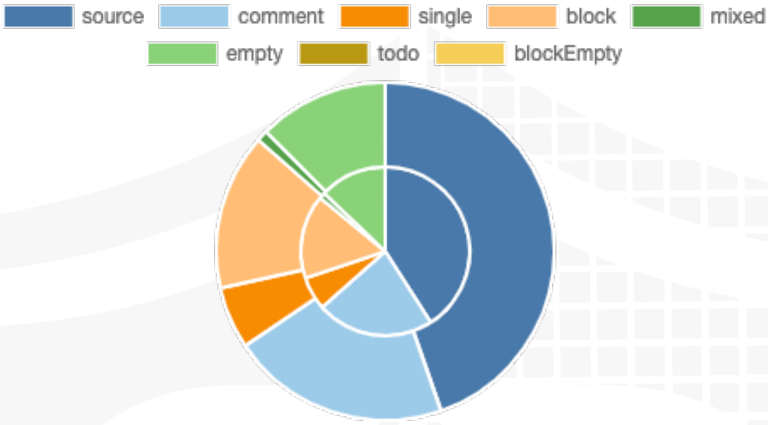


File Name	SHA-1 Hash
contracts/interfaces/IUniswapV2Pair.sol	d9e54a0ef9951d58360e5843e63b24dddf09b746
contracts/interfaces/IExecutionStrategy.sol	cd120bed83d1cbf95518e903b590ba3fecf91c10
contracts/interfaces/IRoyaltyFeeRegistry.sol	1122cdb77a7cd204a360138ebb3f6696c2b0a14e
contracts/interfaces/IHelixmetaExchange.sol	d466298297d7654fcfa5db5d4feb2932e87ba51e
contracts/interfaces/IWETH.sol	36505838b6f5733ba28975e4722dcacd7b8ca853
contracts/interfaces/IHelixmetaToken.sol	65c8ddf91aefcef35bec9d4265028a430e9bac33
contracts/interfaces/ICurrencyManager.sol	d7ff92d4c7bfb37bf77ffa5a38c8844e26b72171
contracts/interfaces/ITransferManagerNFT.sol	3c632f35069e7417b8999f90e513f59d4cac332c
contracts/interfaces/IExecutionManager.sol	b20e9652ba1f88ac5ccf70a168ce91ad5518c17e
contracts/interfaces/IRewardConvertor.sol	3841d9088b5f0dbc6a5d8cab3ee573b3129d30f0
contracts/interfaces/ITransferSelectorNFT.sol	65a884cdb5bbaef917479957b106a34f270592be
contracts/interfaces/IRoyaltyFeeManager.sol	a5acfebd25681a71cd17840d18f662b8d1bdd1dd
contracts/interfaces/IOwnable.sol	67f46dc867664220cbdff4bbb1a3da07c173d2b6
contracts/ExecutionManager.sol	5050c540c4a9d6ce9e46f3d489ce088d5f07d5a0
contracts/Staking/PrivateSaleWithFeeSharing.sol	8ecfd643089b0d573888102f1cc420b9cb3e812d
contracts/Staking/TokenSplitter.sol	01f8cea6a6f49a031ace80e38a306f4509c46614
contracts/Staking/FeeSharingSystem.sol	a1be5c2976eb908ee4a516b9a46c1d6f684d527b
contracts/Staking/TokenDistributor.sol	c8667fec4c7d947fcd4c6d7c087329635f20f744
contracts/Staking/FeeSharingSetter.sol	02515575eec9a1a71b5811c2474c857b1bc57b89
contracts/Staking/StakingPoolForUniswapV2Tokens.sol	b77921d6ae789ebadf5575a1daddc3a5fe8f8b6e
contracts/TransferSelectorNFT.sol	32b48d43ff45860c7a66fbdf1974beac9e001625
contracts/RoyaltyFee/RoyaltyFeeRegistry.sol	bec0a7e3a81df156256b901f05c149ffd1d2d840
contracts/RoyaltyFee/RoyaltyFeeSetter.sol	7f341cd9bff4380c19f0ca156db9736ca562d6e2
contracts/RoyaltyFeeManager.sol	7d39a634d79cfe54517cfca6bceefc666f82f67e
contracts/CurrencyManager.sol	5c75b9ee583c7efa4c3dcaade4e34623ac1ff156
contracts/libraries/SignatureChecker.sol	4ec7323fbfce659769c600904abb6e53721c04c4
contracts/libraries/OrderTypes.sol	2f4766d8720feb779c17f5e8c654192b030a5f2e
contracts/TransferManager/TransferManagerERC721.sol	95669aa816964868ad86a35c8b3ff6abd05dd092
contracts/TransferManager/TransferManagerNonCompliantERC721.sol	b03df79a092838ee715ea39ec1ae7b4ce2202714
contracts/TransferManager/TransferManagerERC1155.sol	ea3a9c893dd8286115f074238ec7d95f4a2bb404
contracts/HelixmetaToken/HelixmetaAirdrop/HelixmetaAirdrop.sol	86c2027394b39218ed7ffb6bee08b3dcbbb8ae02
contracts/HelixmetaToken/HelixmetaToken.sol	585420033caeca37a55874be084c615d42d21489
contracts/HelixmetaExchange.sol	21d87208918b2a3a9ac13d741f24958e29867ffe
contracts/StrategySale/StrategyAnyItemFromCollectionForFixedPrice.sol	ad67ee3d51f51a5fea743e1351441a9a5be17040
contracts/StrategySale/StrategyPrivateSale.sol	cf868736bb2f4c63bc20c80ece96a3e6eff5dd17
contracts/StrategySale/StrategyStandardSaleForFixedPrice.sol	e02cfbcc50890f5b6d9cb46ad876917d45a1f959
contracts/TradingRewardsDistributor.sol	131c77653d4190e281c18ae00b9c27f0eb0f26e2

Metrics

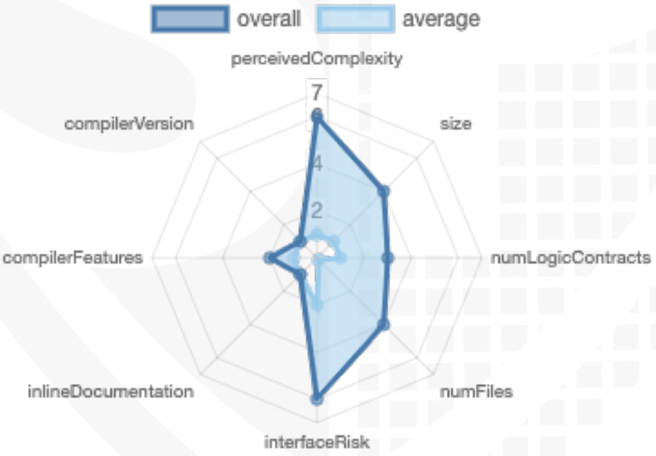
Source Lines

v1.0



Risk Level

v1.0



Capabilities

Components

Version	Contracts	Libraries	Interfaces	Abstract
1.0	22	2	13	0

Exposed Functions

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

Version	Public	Payable
1.0	171	4

Version	External	Internal	Private	Pure	View
1.0	170	152	0	9	70

State Variables

Version	Total	Public
1.0	116	109

Capabilities

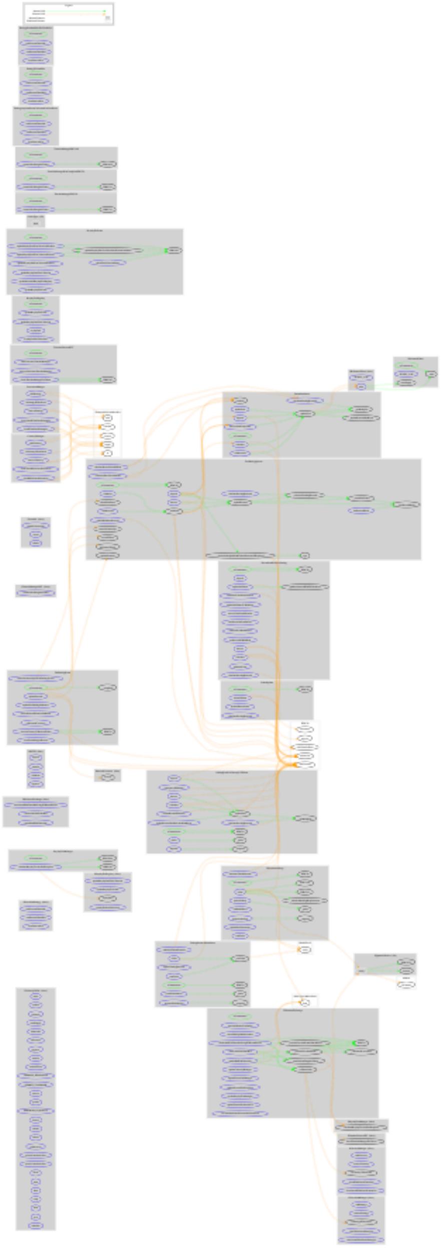
Version	Solidity Versions observed	Experimental Features	Can Receive Funds	Uses Assembly	Has Destroyable Contracts
1.0	`^0.8.0`		`yes`		

Version	Transfers ETH	Low-Level Calls	DelegateCall	Uses Hash Functions	ECRecover	New/ Create/ Create2
1.0				`yes`	`yes`	

Inheritance Graph

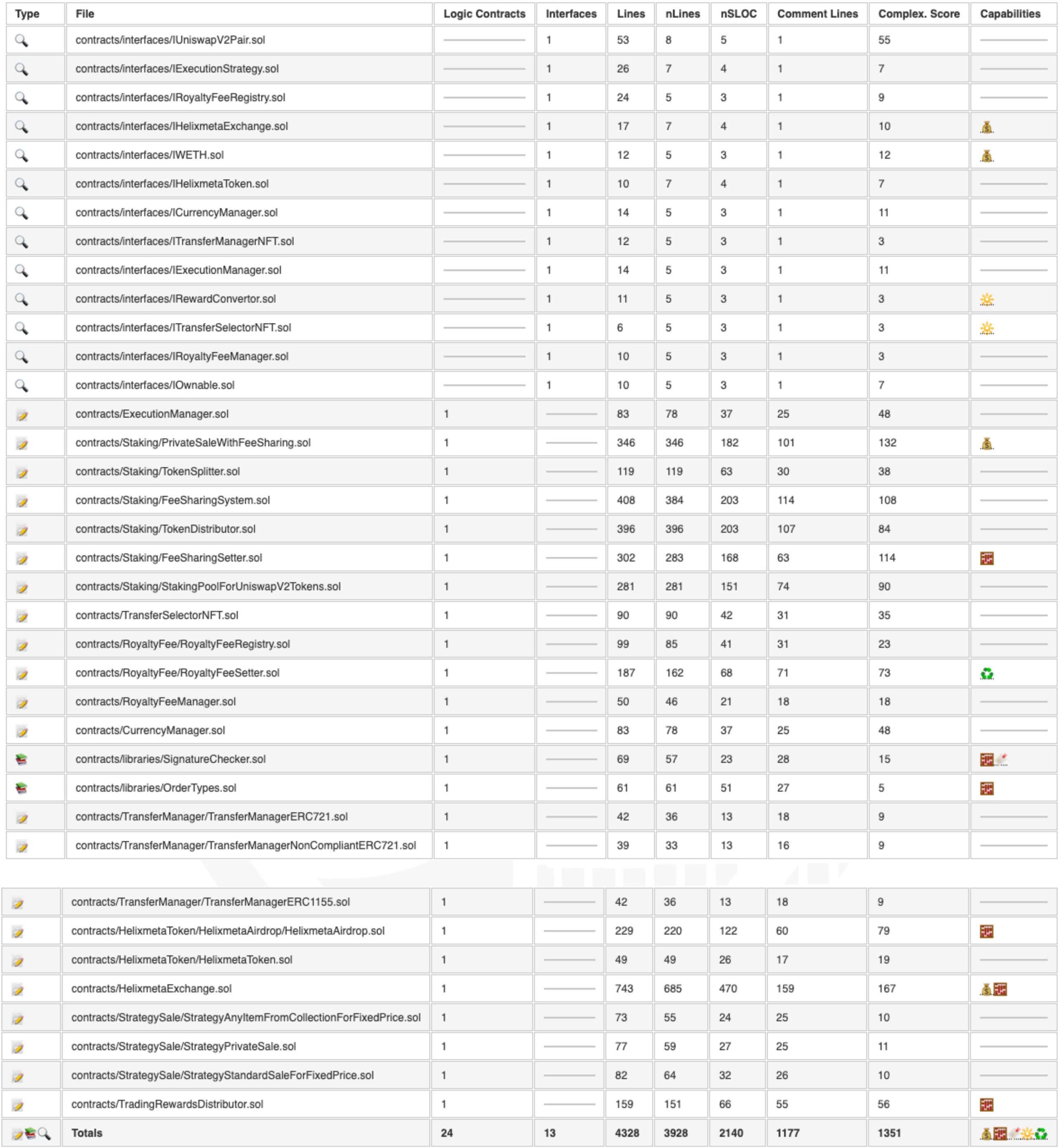
v1.0

CallGraph
v1.0



Source Units in Scope

v1.0



Type	File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
🔍	contracts/interfaces/IUniswapV2Pair.sol	—	1	53	8	5	1	55	—
🔍	contracts/interfaces/IExecutionStrategy.sol	—	1	26	7	4	1	7	—
🔍	contracts/interfaces/IRoyaltyFeeRegistry.sol	—	1	24	5	3	1	9	—
🔍	contracts/interfaces/IHelixmetaExchange.sol	—	1	17	7	4	1	10	💰
🔍	contracts/interfaces/IWETH.sol	—	1	12	5	3	1	12	💰
🔍	contracts/interfaces/IHelixmetaToken.sol	—	1	10	7	4	1	7	—
🔍	contracts/interfaces/ICurrencyManager.sol	—	1	14	5	3	1	11	—
🔍	contracts/interfaces/ITransferManagerNFT.sol	—	1	12	5	3	1	3	—
🔍	contracts/interfaces/IExecutionManager.sol	—	1	14	5	3	1	11	—
🔍	contracts/interfaces/IRewardConvertor.sol	—	1	11	5	3	1	3	🔆
🔍	contracts/interfaces/ITransferSelectorNFT.sol	—	1	6	5	3	1	3	🔆
🔍	contracts/interfaces/IRoyaltyFeeManager.sol	—	1	10	5	3	1	3	—
🔍	contracts/interfaces/IOwnable.sol	—	1	10	5	3	1	7	—
📝	contracts/ExecutionManager.sol	1	—	83	78	37	25	48	—
📝	contracts/Staking/PrivateSaleWithFeeSharing.sol	1	—	346	346	182	101	132	💰
📝	contracts/Staking/TokenSplitter.sol	1	—	119	119	63	30	38	—
📝	contracts/Staking/FeeSharingSystem.sol	1	—	408	384	203	114	108	—
📝	contracts/Staking/TokenDistributor.sol	1	—	396	396	203	107	84	—
📝	contracts/Staking/FeeSharingSetter.sol	1	—	302	283	168	63	114	🧮
📝	contracts/Staking/StakingPoolForUniswapV2Tokens.sol	1	—	281	281	151	74	90	—
📝	contracts/TransferSelectorNFT.sol	1	—	90	90	42	31	35	—
📝	contracts/RoyaltyFee/RoyaltyFeeRegistry.sol	1	—	99	85	41	31	23	—
📝	contracts/RoyaltyFee/RoyaltyFeeSetter.sol	1	—	187	162	68	71	73	♻️
📝	contracts/RoyaltyFeeManager.sol	1	—	50	46	21	18	18	—
📝	contracts/CurrencyManager.sol	1	—	83	78	37	25	48	—
📚	contracts/libraries/SignatureChecker.sol	1	—	69	57	23	28	15	🧮🔖
📚	contracts/libraries/OrderTypes.sol	1	—	61	61	51	27	5	🧮
📝	contracts/TransferManager/TransferManagerERC721.sol	1	—	42	36	13	18	9	—
📝	contracts/TransferManager/TransferManagerNonCompliantERC721.sol	1	—	39	33	13	16	9	—
📝	contracts/TransferManager/TransferManagerERC1155.sol	1	—	42	36	13	18	9	—
📝	contracts/HelixmetaToken/HelixmetaAirdrop/HelixmetaAirdrop.sol	1	—	229	220	122	60	79	🧮
📝	contracts/HelixmetaToken/HelixmetaToken.sol	1	—	49	49	26	17	19	—
📝	contracts/HelixmetaExchange.sol	1	—	743	685	470	159	167	💰🧮
📝	contracts/StrategySale/StrategyAnyItemFromCollectionForFixedPrice.sol	1	—	73	55	24	25	10	—
📝	contracts/StrategySale/StrategyPrivateSale.sol	1	—	77	59	27	25	11	—
📝	contracts/StrategySale/StrategyStandardSaleForFixedPrice.sol	1	—	82	64	32	26	10	—
📝	contracts/TradingRewardsDistributor.sol	1	—	159	151	66	55	56	🧮
📝📚🔍	**Totals**	**24**	**13**	**4328**	**3928**	**2140**	**1177**	**1351**	💰🧮🔖🔆♻️

Legend

Attribute	Description
Lines	total lines of the source unit
nLines	normalized lines of the source unit (e.g. normalizes functions spanning multiple lines)

nSLOC	normalized source lines of code (only source-code lines; no comments, no blank lines)
Comment Lines	lines containing single or block comments
Complexity Score	a custom complexity score derived from code statements that are known to introduce code complexity (branches, loops, calls, external interfaces, ...)

Audit Results

AUDIT PASSED

Critical issues

No critical issues

High issues

No high issues

Medium issues

No medium issues

Low issues

Issue	File	Type	Line	Description
#1	All files	A floating pragma is set	At the top of source files	The current pragma Solidity directive is „“^0.8.0””.
#2	HelixmetaAirdrop	Missing Zero Address Validation (missing-zero-check)	66, 65, 64, 67,	Check that the address is not zero
#3	HelixmetaExchange	Missing Zero Address Validation (missing-zero-check)	97, 98, 427	Check that the address is not zero
#4	RoyaltyFeeSetter	Missing Zero Address Validation (missing-zero-check)	31	Check that the address is not zero
#5	TokenDistributor	Missing Zero Address Validation (missing-zero-check)	89	Check that the address is not zero

#6	TransferManagerERC1155	Missing Zero Address Validation (missing-zero-check)	19	Check that the address is not zero
#7	TransferManagerERC721	Missing Zero Address Validation (missing-zero-check)	19	Check that the address is not zero
#8	TransferManagerNonCompliantERC721	Missing Zero Address Validation (missing-zero-check)	18	Check that the address is not zero
#9	TransferSelectorNFT	Missing Zero Address Validation (missing-zero-check)	36	Check that the address is not zero
#10	RoyaltyFeeSetter	Local variables shadowing	116	Rename the local variables that shadow another component

Informational issues

Issue	File	Type	Line	Description
#1	FeeSharingSystem	State variables that could be declared constant (constable-states)	37	Add the `constant` attributes to state variables that never change

Audit Comments

We recommend you to use the special form of comments (NatSpec Format, Follow link for more information https://docs.soliditylang.org/en/v0.5.10/natspec-format.html) for your contracts to provide rich documentation for functions, return variables and more. This helps investors to make clear what that variables, functions etc. do.

30. April 2022:

- Read whole report for more information

SWC Attacks

ID	Title	Relationships	Status
SWC-136	Unencrypted Private Data On-Chain	CWE-767: Access to Critical Private Variable via Public Method	PASSED
SWC-135	Code With No Effects	CWE-1164: Irrelevant Code	PASSED
SWC-134	Message call with hardcoded gas amount	CWE-655: Improper Initialization	PASSED
SWC-133	Hash Collisions With Multiple Variable Length Arguments	CWE-294: Authentication Bypass by Capture-replay	PASSED
SWC-132	Unexpected Ether balance	CWE-667: Improper Locking	PASSED
SWC-131	Presence of unused variables	CWE-1164: Irrelevant Code	PASSED
SWC-130	Right-To-Left-Override control character (U+202E)	CWE-451: User Interface (UI) Misrepresentation of Critical Information	PASSED
SWC-129	Typographical Error	CWE-480: Use of Incorrect Operator	PASSED
SWC-128	DoS With Block Gas Limit	CWE-400: Uncontrolled Resource Consumption	PASSED

SWC-127	Arbitrary Jump with Function Type Variable	CWE-695: Use of Low-Level Functionality	PASSED
SWC-125	Incorrect Inheritance Order	CWE-696: Incorrect Behavior Order	PASSED
SWC-124	Write to Arbitrary Storage Location	CWE-123: Write-what-where Condition	PASSED
SWC-123	Requirement Violation	CWE-573: Improper Following of Specification by Caller	PASSED
SWC-122	Lack of Proper Signature Verification	CWE-345: Insufficient Verification of Data Authenticity	PASSED
SWC-121	Missing Protection against Signature Replay Attacks	CWE-347: Improper Verification of Cryptographic Signature	PASSED
SWC-120	Weak Sources of Randomness from Chain Attributes	CWE-330: Use of Insufficiently Random Values	PASSED
SWC-119	Shadowing State Variables	CWE-710: Improper Adherence to Coding Standards	NOT PASSED
SWC-118	Incorrect Constructor Name	CWE-665: Improper Initialization	PASSED
SWC-117	Signature Malleability	CWE-347: Improper Verification of Cryptographic Signature	PASSED

SWC-116	Timestamp Dependence	CWE-829: Inclusion of Functionality from Untrusted Control Sphere	PASSED
SWC-115	Authorization through tx.origin	CWE-477: Use of Obsolete Function	PASSED
SWC-114	Transaction Order Dependence	CWE-362: Concurrent Execution using Shared Resource with Improper Synchronization ('Race Condition')	PASSED
SWC-113	DoS with Failed Call	CWE-703: Improper Check or Handling of Exceptional Conditions	PASSED
SWC-112	Delegatecall to Untrusted Callee	CWE-829: Inclusion of Functionality from Untrusted Control Sphere	PASSED
SWC-111	Use of Deprecated Solidity Functions	CWE-477: Use of Obsolete Function	PASSED
SWC-110	Assert Violation	CWE-670: Always-Incorrect Control Flow Implementation	PASSED
SWC-109	Uninitialized Storage Pointer	CWE-824: Access of Uninitialized Pointer	PASSED
SWC-108	State Variable Default Visibility	CWE-710: Improper Adherence to Coding Standards	PASSED
SWC-107	Reentrancy	CWE-841: Improper Enforcement of Behavioral Workflow	PASSED
SWC-106	Unprotected SELFDESTRUCT Instruction	CWE-284: Improper Access Control	PASSED

SWC-105	Unprotected Ether Withdrawal	CWE-284: Improper Access Control	PASSED
SWC-104	Unchecked Call Return Value	CWE-252: Unchecked Return Value	PASSED
SWC-103	Floating Pragma	CWE-664: Improper Control of a Resource Through its Lifetime	NOT PASSED
SWC-102	Outdated Compiler Version	CWE-937: Using Components with Known Vulnerabilities	PASSED
SWC-101	Integer Overflow and Underflow	CWE-682: Incorrect Calculation	PASSED
SWC-100	Function Default Visibility	CWE-710: Improper Adherence to Coding Standards	PASSED

Solid Proofed

Blockchain Security | Smart Contract Audits | KYC

MADE IN GERMANY