SOLIDProof

Bring trust into your projects

Blockchain Security | Smart Contract Audits | KYC Development | Marketing

MADE IN GERMANY

Company DAO

Audit

Security Assessment

19. May, 2023

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	21. March 2023	• Layout project • Automated- /Manual-Security Testing • Summary
1.1	28. March 2023	• Reaudit
1.2	27. April 2023	• Added functionality audit
1.3	19. May 2023	• Added functionality re-audit

Network
Ethereum (ERC20)

Website
http://companydao.org

Reddit
https://www.reddit.com/user/company_dao/

LinkedIn
http://www.linkedin.com/in/taylorneff
https://www.linkedin.com/company/delta-alpha-omega-llc

Description

Company DAO is a web3 protocol which links legal entities in multiple jurisdictions to immutable /smart contracts/ on the blockchain. This allows real world companies and assets to be legally owned and managed by online communities through a /decentralised autonomous organisation/, whereby the smart contracts act as a single source of truth for decisions made by the company's stakeholders.

Project Engagement

During the 20th of March 2023, **CompanyDAO 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/CompanyDAO/protocol-contracts
  - Commit:
    - dd3e683

v1.1

https://github.com/CompanyDAO/protocol-contracts/commit/ff1503ad3fce81c8e1ad8d09c50b8bb8a8f6af8f

v1.2

https://github.com/CompanyDAO/protocol-contracts/commit/5a5e02cc320630f55728d699d235b8e795c1ac80

v1.3

https://github.com/CompanyDAO/protocol-contracts/commit/8c40b3749647719233adac41949fb000c761566e

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-upgradeable/access/AccessControlEnumerableUpgradeable.sol	3
@openzeppelin/contracts-upgradeable/access/IAccessControlEnumerableUpgradeable.sol	1
@openzeppelin/contracts-upgradeable/access/OwnableUpgradeable.sol	4
@openzeppelin/contracts-upgradeable/governance/utils/IVotesUpgradeable.sol	1
@openzeppelin/contracts-upgradeable/proxy/utils/Initializable.sol	9
@openzeppelin/contracts-upgradeable/proxy/utils/UUPSUpgradeable.sol	2
@openzeppelin/contracts-upgradeable/security/PausableUpgradeable.sol	2
@openzeppelin/contracts-upgradeable/security/ReentrancyGuardUpgradeable.sol	5
@openzeppelin/contracts-upgradeable/token/ERC20/IERC20Upgradeable.sol	4
@openzeppelin/contracts-upgradeable/token/ERC20/extensions/ERC20CappedUpgradeable.sol	1
@openzeppelin/contracts-upgradeable/token/ERC20/extensions/ERC20VotesUpgradeable.sol	1
@openzeppelin/contracts-upgradeable/token/ERC20/utils/SafeERC20Upgradeable.sol	4
@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol	3
@openzeppelin/contracts-upgradeable/utils/Create2Upgradeable.sol	1
@openzeppelin/contracts-upgradeable/utils/math/MathUpgradeable.sol	3
@openzeppelin/contracts-upgradeable/utils/structs/EnumerableSetUpgradeable.sol	2
@openzeppelin/contracts/proxy/beacon/BeaconProxy.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.

v1.2

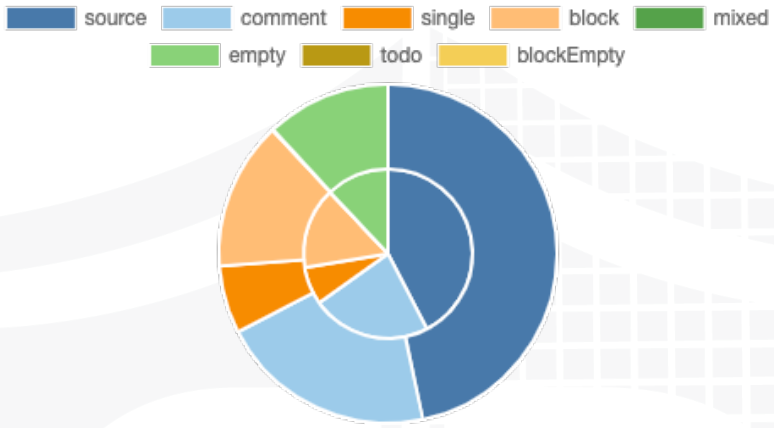
File Name	SHA-1 Hash
contracts/interfaces/IVesting.sol	8891d9ecb49896e11c6b29aec102e14fbae91beb
contracts/interfaces/IInvoice.sol	66e679ad6f61ae4fdba0bfc0503691f4f55252dd
contracts/interfaces/ITokenFactory.sol	35fc9a89369bbb3bf21b31f489f624384cae8348
contracts/interfaces/registry/ICompaniesRegistry.sol	e1f168670c06cb13a12401e4ec330f67ca91722e
contracts/interfaces/registry/IRegistry.sol	4b5247f2c4cac77da0cee3cae4bc4ca7a43f5b8b
contracts/interfaces/registry/ITokensRegistry.sol	e3850a5d6bab5409d2ac5056da5e2da343404b73
contracts/interfaces/registry/IRecordsRegistry.sol	5dfdba1cc302553a07c5986d0232d2673834f4d9
contracts/interfaces/ITGE.sol	5faabd6888c3576d425cea0118e108b4f6422625
contracts/interfaces/ITGEFactory.sol	3e25b2a13fc80a98244c652f17f871aa7719a303
contracts/interfaces/IPool.sol	2cff7eae5a34de725f02ca89c3d59caccf1c6527
contracts/interfaces/IService.sol	8e9cf7722ba6438ab44f83bd7822987d06d655cb

contracts/interfaces/IToken.sol	d47ed31f31c398584d8925bab71ad3c5740106b4
contracts/interfaces/governor/IGovernorProposals.sol	fa666462a35e3d43c66498188ecfc2fd075c1269
contracts/interfaces/governor/IGovernor.sol	00086917df60f3e4d98701c87d4b9a435595e51b
contracts/interfaces/governor/IGovernanceSettings.sol	92d3739c68933a0618f77425573b939b510f8dff
contracts/interfaces/ICustomProposal.sol	4e4733b3c8d79c027b9ee6ddaa481d26c470e645
contracts/TokenFactory.sol	489f0d3df85e08bf8321598641f21485c6c1fa40
contracts/TokenERC1155.sol	da39a3ee5e6b4b0d3255bfef95601890afd80709
contracts/Invoice.sol	41981fdd38602caff2f0ee95740f1acd80d9f270
contracts/registry/RegistryBase.sol	2a33c6b38aff397bf063f81662c8c86b9eb4e604
contracts/registry/CompaniesRegistry.sol	8beb2e879543cb49f6a93c103a35b0f7fd415111
contracts/registry/TokensRegistry.sol	2f6319af971cd106843590d61f9cf93f0549ca3d
contracts/registry/RecordsRegistry.sol	79be33ecb65ce697723fcc618218744a0c0f5e81
contracts/Vesting.sol	7605936664f25713434205e1a499bff670dfaca1
contracts/Registry.sol	fee7cbcda25460a62369afb3d3d72ab10d543374
contracts/libraries/ExceptionsLibrary.sol	38e97c0e17fa0cc2ad0e8e6d6ef313171bbf7287

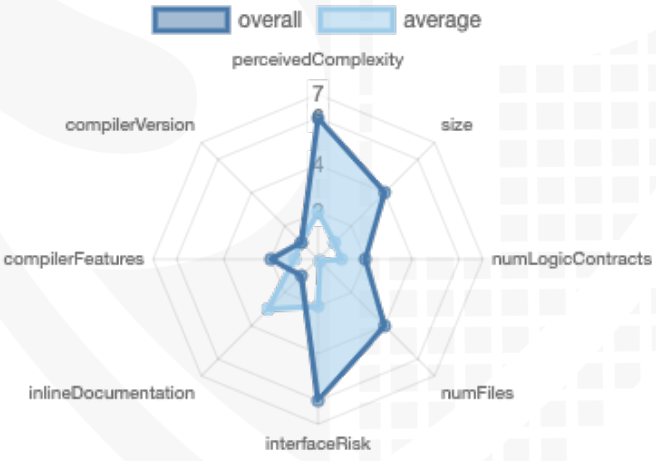
contracts/Service.sol	fa1f044eba9a09d218b33115d0b03eb8c1166eb3
contracts/TGEFactory.sol	fcd16b1c2790a538a5bb33a814c78bb96139ea73
contracts/Pool.sol	1fe8d034fff6e6815bdd6b587ea923d93b09b3c4
contracts/TGE.sol	bbd4d35920a11211600f471383e3c3f9d7b4c8db
contracts/governor/GovernanceSettings.sol	6f1a0095f0592e4b86ebe84dba1228c28bc5028b
contracts/governor/Governor.sol	243e6ba42f2ffcc4546a4ae2eea9f488df5e0d23
contracts/governor/GovernorProposals.sol	b19c67266b8018de35b29f26636e01eb2ae97c3d
contracts/CustomProposal.sol	bb688979346cbbd472f0128f02f3593675fc7255
contracts/Token.sol	437c5247fbb5e43deb1ad27f3e341bb0a44c55ac

Metrics

Source Lines
v1.0



Risk Level
v1.0



Capabilities

Components

Version	Contracts	Libraries	Interfaces	Abstract
1.2	10	1	16	7

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
1.2	241	4

Version	External	Internal	Private	Pure	View
1.0	187	176	4	7	128

State Variables

Version	Total	Public
1.0	158	144

Capabilities

Version	Solidity Versions observed	Experimental Features	Can Receive Funds	Uses Assembly	Has Destroyable Contracts
1.0	`0.8.17`		`yes`		

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

1.0				`yes`		`yes → NewContract:BeaconProxy`

Inheritance Graph
v1.2

CallGraph
v1.2

Scope of Work/Verify Claims

The above token Team provided us with the files that needs to be tested (Github, Bscscan, Etherscan, files, etc.). The scope of the audit is the main contract (usual the same name as team appended with .sol).

We will verify the following claims:

1. Are the contracts upgradeable
2. Overall checkup (Smart Contract Security)

Are the contracts upgradeable

Name	
Is contract an upgradeable?	Yes

Comments:

v1.0

- Owner can deploy a new version of the contracts which can change any limit and give owner new privileges
    - Be aware of this and do your own research for the contract which is the contract pointing to

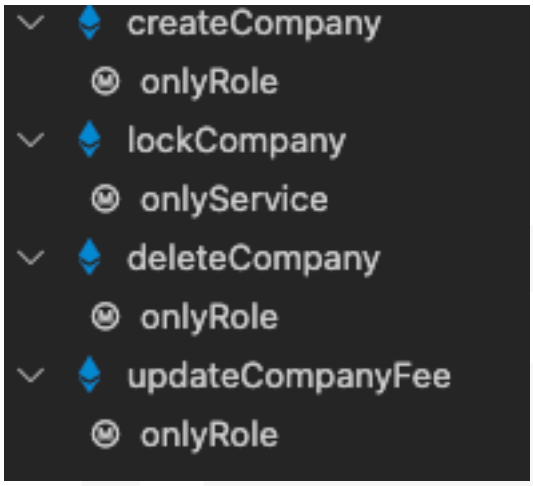
Legend

Attribute	Symbol
Verified / Checked	✓
Partly Verified	⚑
Unverified / Not checked	✗
Not available	–

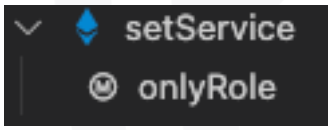
Modifiers and public functions

v1.3

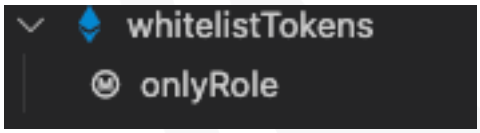
Registry/CompaniesRegistry



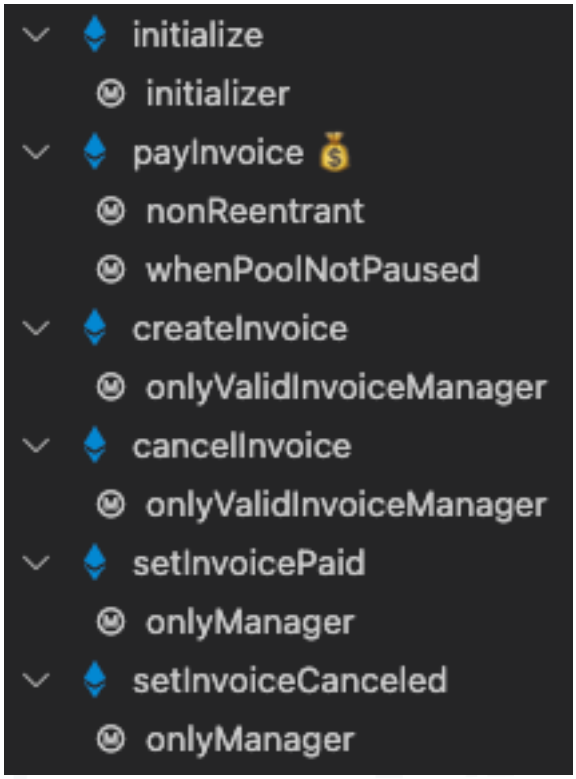
Registry/RegistryBase



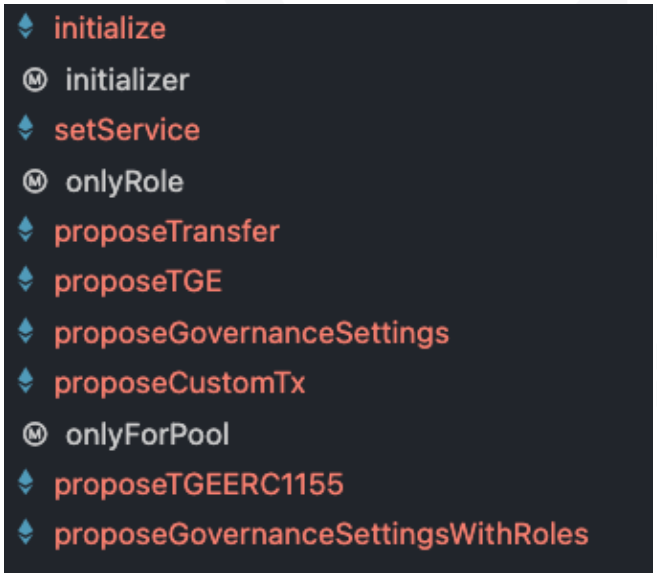
Registry/TokensRegistry



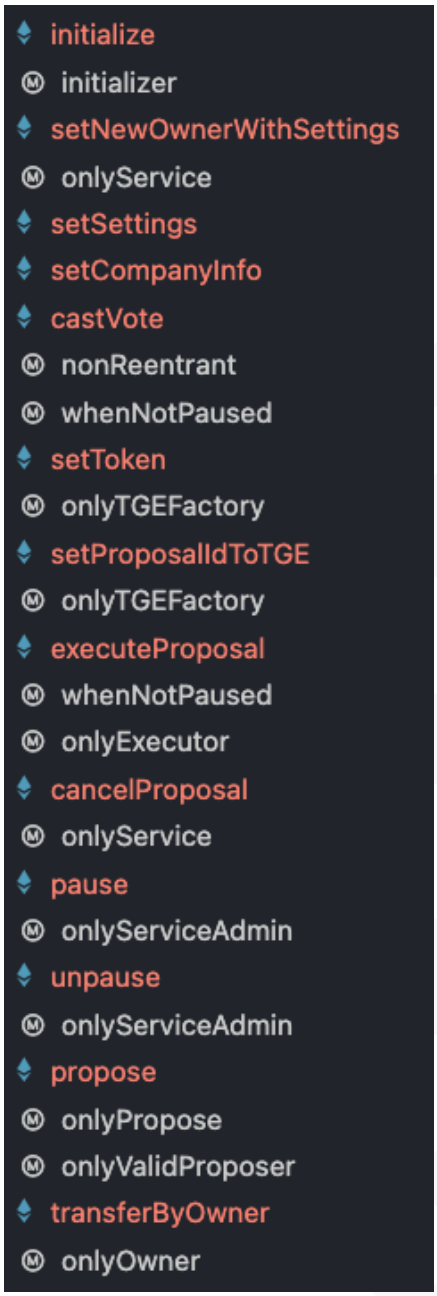
Invoice



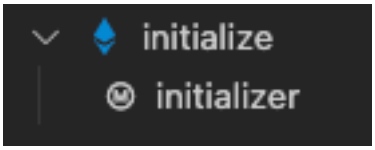
CustomProposal



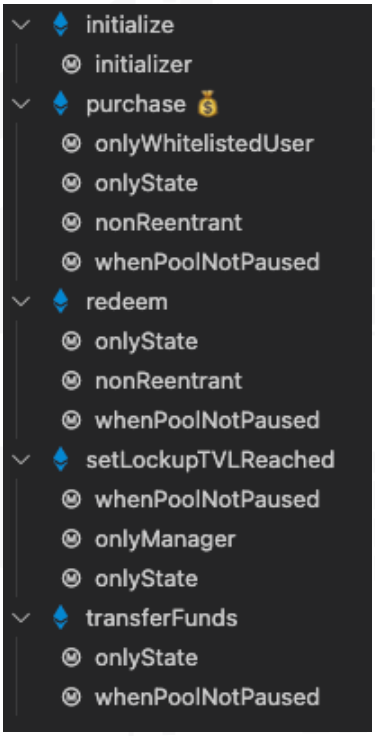
Pool



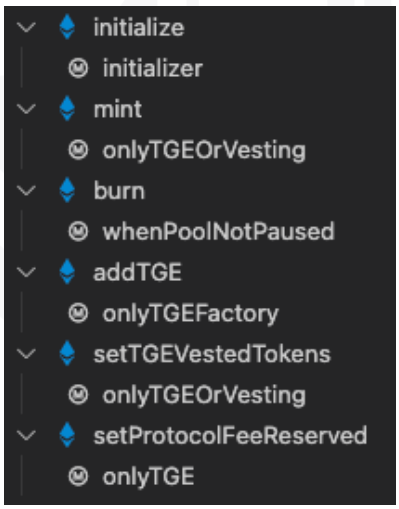
Registry



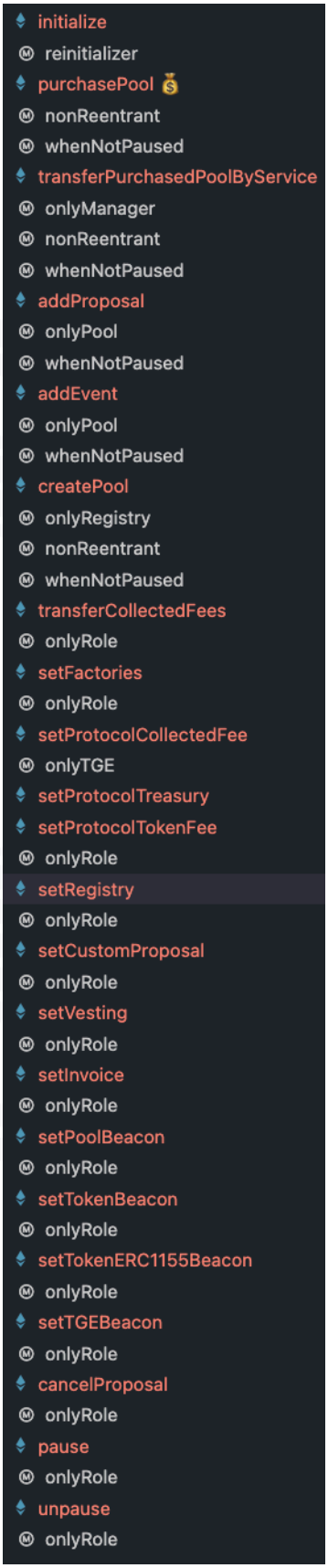
TGE



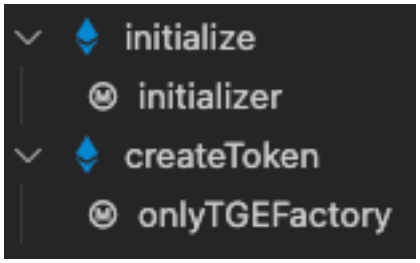
Token



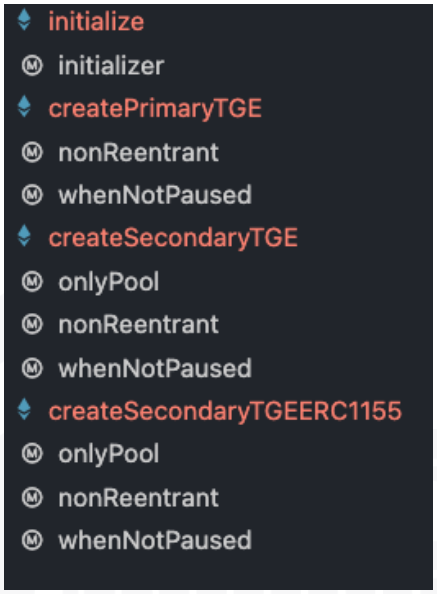
Service



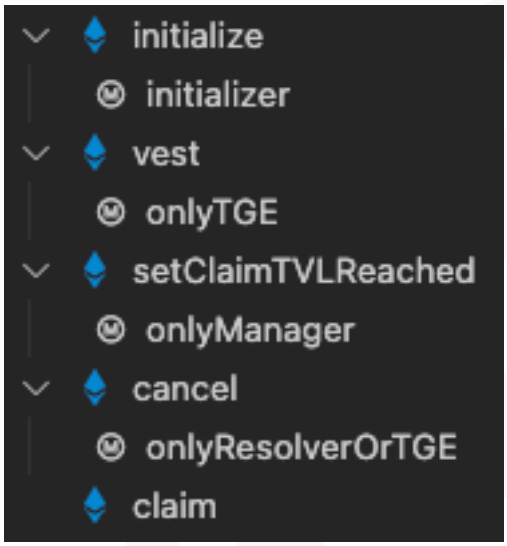
TokenFactory



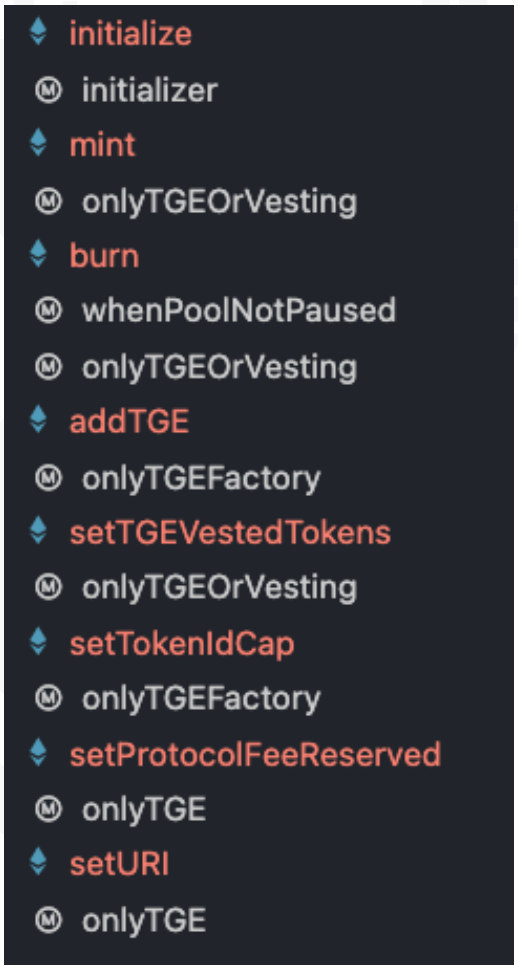
TGEFactory



Vesting



TokenERC1155



Note: Imported contracts from official packages were not listed down below

Comments

- CompaniesRegistry
  - Companies manager role is able to
    - Delete company
    - Update company fee to any arbitrary value including 100% or more
    - Lock company
    - Create company
- RecordsRegistry
  - Only service or factory is able to
    - Add contract record
    - Add event record
  - Only service is able to
    - Add proposal
- RegistryBase
  - Only default admin is able to
    - change service address
- TokenRegistry
  - Only default admin is able to
    - Grant whitelist role to token
- CustomProposal
  - Initializer will be the default admin
  - Only default admin is able to
    - change service address
  - Only 'Pool' role address can propose a custom transactions
- Invoice
  - Only manager is able to
    - set invoice status
      - cancel
      - paid
  - Only valid invoice manager is able to
    - set invoice status
      - cancel
      - paid
    - Create invoice
  - PayInvoice function can be locked by pausing pool
  - While calling the initialize the registry can be set
- Pool
  - Only owner is able to
    - Transfers funds from treasury if the pool is not dao
  - Only pool is able to
    - Change pool executor

        - Change pool secretary
    - Only propose is able to
        - Create a proposal
    - Only service admin is able to
        - Pause/unpause contract
    - Only Service is able to
        - Set New Owner with settings
        - Cancel Proposal
    - Only executor is able to
        - Execute proposal
    - Only TGEFactory is able to
        - Add a new entry about the deployed token contract to the list of tokens related to the pool
        - Set proposal ID

    - While initializing the service and the company info will be set
- Registry
    - While initializing the caller will be the default admin
- Service
    - Only default admin is able to
        - Pause/unpause
        - Cancel proposal
        - Update
            - tge beacon
            - Token beacon
            - Pool Beacon
            - Token and TGE factory address
            - ERC1155 Token beacon address
            - Invoice
            - Vesting
            - Custom proposal
            - Registry
            - Protol token fee
            - Protocol treasury address
            - Protocol collected fee address
        - Transfer collected fees to an arbitrary address
    - Only registry is able to
        - Create a pool
    - Only pool is able to
        - Add new event to directory
        - Add new proposal record
    - Only Manager Role address is able to create a new pool
- TGE
    - Only manager is able to
        - lockup tvl reached

        - Only whitelisted user is able to
            - Purchase pool tokens
- TGEFactory
    - Only pool is able to
        - Create secondary TGE
        - Create primary TGE
        - Create secondary TGE ERC1155
- Token
    - Only TGE is able to
        - Set total protocol fee reserved without any limitations
    - Only TGE or vesting is able to
        - Total vested amount without any limitations
        - Mint new tokens
        - Burn tokens from any address
    - Only TGE Factory is able tot
        - Add tge addresses
- TokenFactory
    - Only TGE Factory is able to
        - Create new token contract
- Vesting
    - Only TGE is able to
        - Vest token to an arbitrary address
    - Only Manager is able to
        - Claim tvl reached for TGE
    - Only Resolver or tge is able to
        - Cancel vesting
- TokenERC1155
    - Only TGE or Vesting contract addresses are able to
        - Mint tokens
        - Burn tokens from any address
        - Set any arbitrary amount of tokens to total vested tokens
    - Only TGE factory contract addresses are able to
        - Add TGE address
        - Set token ID cap to any arbitrary value
    - Only TGE contract addresses are able to
        - Set URI
        - Set the amount of reserved tokens for minting protocol fee to any arbitrary value
- There are several authorities which are authorized to call some functions, that means, if the owner is renounced, another address is still authorized to call functions
    - Be aware of this

Please check if an OnlyOwner or similar restrictive modifier has been forgotten.

Source Units in Scope
v1.2

File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score
contracts/interfaces/IVesting.sol		1	36	17	13	1	13
contracts/interfaces/IInvoice.sol		1	28	28	24	1	1
contracts/interfaces/ITokenFactory.sol		1	13	8	4	1	3
contracts/interfaces/registry/ICompaniesRegistry.sol		1	21	17	12	1	3
contracts/interfaces/registry/IRegistry.sol		1	12	11	7	1	9
contracts/interfaces/registry/ITokensRegistry.sol		1	7	6	3	1	3
contracts/interfaces/registry/IRecordsRegistry.sol		1	82	62	33	22	9
contracts/interfaces/ITGE.sol		1	52	30	23	1	19
contracts/interfaces/ITGEFactory.sol		1	15	9	5	1	3
contracts/interfaces/IPool.sol		1	78	12	8	1	43
contracts/interfaces/IService.sol		1	84	17	13	1	51
contracts/interfaces/IToken.sol		1	67	25	19	1	43
contracts/interfaces/governor/IGovernorProposals.sol		1	9	8	4	1	3
contracts/interfaces/governor/IGovernor.sol		1	42	38	17	16	3
contracts/interfaces/governor/IGovernanceSettings.sol		1	28	25	12	10	3
contracts/interfaces/ICustomProposal.sol		1	5	5	2	1	1
contracts/TokenFactory.sol	1		66	62	33	17	37
contracts/TokenERC1155.sol			1	1			
contracts/Invoice.sol	1		297	268	147	73	97
contracts/registry/RegistryBase.sol	1		45	43	25	6	17
contracts/registry/CompaniesRegistry.sol	1		246	218	90	91	58
contracts/registry/TokensRegistry.sol	1		36	34	16	10	20
contracts/registry/RecordsRegistry.sol	1		195	169	77	74	39
contracts/Vesting.sol	1		278	262	127	92	81
contracts/Registry.sol	1		56	49	19	21	15
contracts/libraries/ExceptionsLibrary.sol	1		74	74	71	1	66
contracts/Service.sol	1		733	662	311	251	264
contracts/TGEFactory.sol	1		375	338	211	84	142
contracts/Pool.sol	1		574	508	289	145	250
contracts/TGE.sol	1		553	523	289	149	193
contracts/governor/GovernanceSettings.sol	1		124	117	58	34	21
contracts/governor/Governor.sol	1		484	408	239	143	58
contracts/governor/GovernorProposals.sol	1		47	47	31	9	12
contracts/CustomProposal.sol	1		615	555	374	117	386
contracts/Token.sol	1		397	359	176	127	132
Totals	**18**	**16**	**5775**	**5015**	**2782**	**1505**	**2098**

Legend

Attribute	Description
Lines	total lines of the source unit
nLines	normalised lines of the source unit (e.g. normalises functions spanning multiple lines)
nSLOC	normalised 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

Critical issues

No critical issues

High issues

No high issues

Medium issues

Issue	File	Type	Line	Description
#1	Service.sol	Owner can set fees up to 100%	443	The address with the “DEFAULT_ADMIN_ROLE” can set the fees up to a 100% which is not recommended. Because it may result in loss of user funds.

Low issues

Issue	File	Type	Line	Description
#1	CustomProposal.sol	Missing Zero Address Validation (missing-zero-check)	76	Check that the address is not zero
#2	Vesting.sol	Missing Zero Address Validation (missing-zero-check)	131, 142	Check that the address is not zero
#3	Token.sol	Missing Events Arithmetic	All	Emit an event for critical parameter changes
#4	TokenERC1155.sol	Missing Events Arithmetic	All	Emit an event for critical parameter changes

Informational issues

No informational issues

Alleviation

The medium bug has been acknowledged by the CompanyDAO team as part of the intended behaviour for their project.

Audit Comments

We recommend you to use the special form of comments (NatSpec Format, Follow link for more information https://docs.soliditylang.org/en/latest/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.

19. May 2023:

- There is still an owner (Owner still has not renounced ownership)
- Owner can deploy a new version of the contract which can change any limit and give owner new privileges
- The pool contract can receive ETH, and the owner can withdraw it before the DAO has started.
- Read whole report and modifiers section 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	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	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