SOLIDProof
Bring trust into your projects

Blockchain Security | Smart Contract Audits | KYC

MADE IN GERMANY

DGNX

Audit

Security Assessment
23.August,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	19.August,2022	• Layout project • Automated- /Manual-Security Testing • Summary

Network

Binance (BSC)

Website
https://dgnx.finance/

Twitter
https://twitter.com/degenecosystem

Telegram
https://t.me/degentraderstoken

Instagram
https://instagram.com/degenecosystem

Discord
https://discord.gg/KWX3kmtX

Facebook
https://www.facebook.com/people/Degen-Trader/100078427221036/

TikTok
https://www.tiktok.com/@degen_traders

OpenSea
https://opensea.io/collection/thedegentrader

RaritySniffer
https://raritysniffer.com/viewcollection/degentraders

Description

DegenX is multichain ecosystem that offers a suite of decentralized applications (dApps) and services to provide solutions for projects and individuals in the DeFi space.

Project Engagement

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

Logo

Contract Links
v1.0
   Provided as files

Vulnerability & Risk Level

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

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

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 analyzing 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/Ownable.sol	6
@openzeppelin/contracts/security/Pausable.sol	1
@openzeppelin/contracts/security/ReentrancyGuard.sol	5
@openzeppelin/contracts/token/ERC20/ERC20.sol	5
@openzeppelin/contracts/token/ERC20/extensions/ERC20Burnable.sol	1
@openzeppelin/contracts/token/ERC20/extensions/ERC20Snapshot.sol	1
@openzeppelin/contracts/token/ERC20/extensions/ERC20Votes.sol	1
@openzeppelin/contracts/token/ERC20/extensions/IERC20Metadata.sol	1
@openzeppelin/contracts/token/ERC20/extensions/draft-ERC20Permit.sol	1
@openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol	3
@openzeppelin/contracts/token/ERC721/ERC721.sol	1
@openzeppelin/contracts/token/ERC721/IERC721.sol	1
@openzeppelin/contracts/token/ERC721/extensions/ERC721Enumerable.sol	1
@openzeppelin/contracts/utils/Address.sol	2
@openzeppelin/contracts/utils/Counters.sol	1
@openzeppelin/contracts/utils/Strings.sol	1
@openzeppelin/contracts/utils/math/SafeMath.sol	3
@uniswap/lib/contracts/libraries/TransferHelper.sol	1
@uniswap/v2-core/contracts/interfaces/IUniswapV2Factory.sol	2
@uniswap/v2-core/contracts/interfaces/IUniswapV2Pair.sol	2

Tested Contract Files

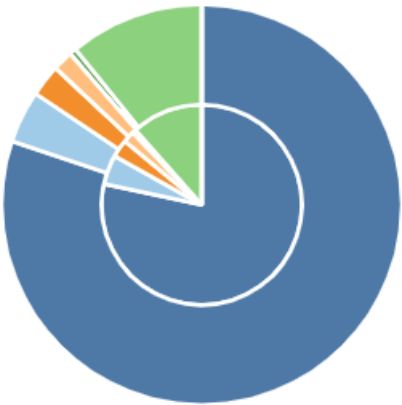This audit covered the following files listed below with a SHA-1 Hash.

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.
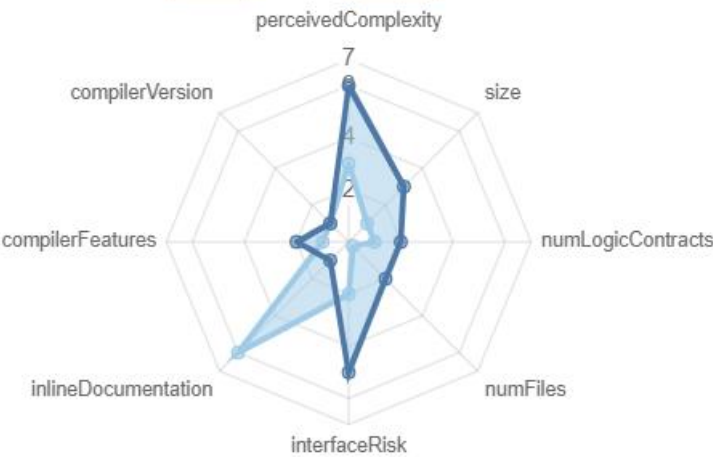
v1.0

File Name	SHA-1 Hash
contracts/interfaces/IDGNXPrivateSale NFT.sol	0e70717295fe075c58ce73b24e6684c4a7 0f0fe4
contracts/interfaces/IDGNXController.s ol	1d4459057fd9ed19ee6c44221252a39e0 7f54d88
contracts/dgnx/DGNXSale.sol	2c9fe45f9aef01c5965f1894fe18e3f56a5bf 1b2
contracts/dgnx/DGNX.sol	c6625f3002fa62ac4c4430b245cc5fd35b2 46761
contracts/dgnx/DGNXPrivateSaleNFT.s ol	82c84281daf7263693d94937c6964fb5b3 1ee406
contracts/dgnx/DGNXLibrary.sol	b83d7359f4ee51dc9f893ffb6b056b0c73f6 4310
contracts/dgnx/DGNXLocker.sol	30805a01ab5f50eb6923f2ad0f8f7e39474 5f92f
contracts/dgnx/DGNXController.sol	7e1053c64f6a69aadabd320dd1c1f4b93fd c8463
contracts/dgnx/DGNXLegacyDisburser. sol	a0490337c92ec4cde2b9e7bebbc13815d 376f7d5

Metrics

Source Lines
v1.0

Risk Level
v1.0

Capabilities

v1.0

Components

📝 Contracts	📚 Libraries	🔍 Interfaces	🎨 Abstract
6	1	2	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.

🌐 Public	💰 Payable
98	6

External	Internal	Private	Pure	View
79	111	3	3	31

StateVariables

Total	🌐 Public
81	59

Capabilities

Solidity Versions observed	🖋 Experimental Features	💰 Can Receive Funds	📑 Uses Assembly	🔴 Has Destroyable Contracts

`^0.8.0`
`^0.8.4`
`^0.8.13`
`>=0.8.4`
		yes		

📤 Transfers ETH	⚡ Low-Level Calls	👥 DelegateCall	📑 Uses Hash Functions	⚡ ECRecover	🌀 New/Create/Create2
yes					

♻ TryCatch	Σ Unchecked

Inheritance Graph
v1.0
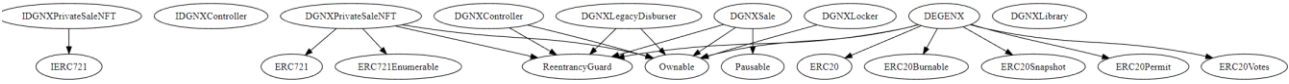
Call Graph
v1.0
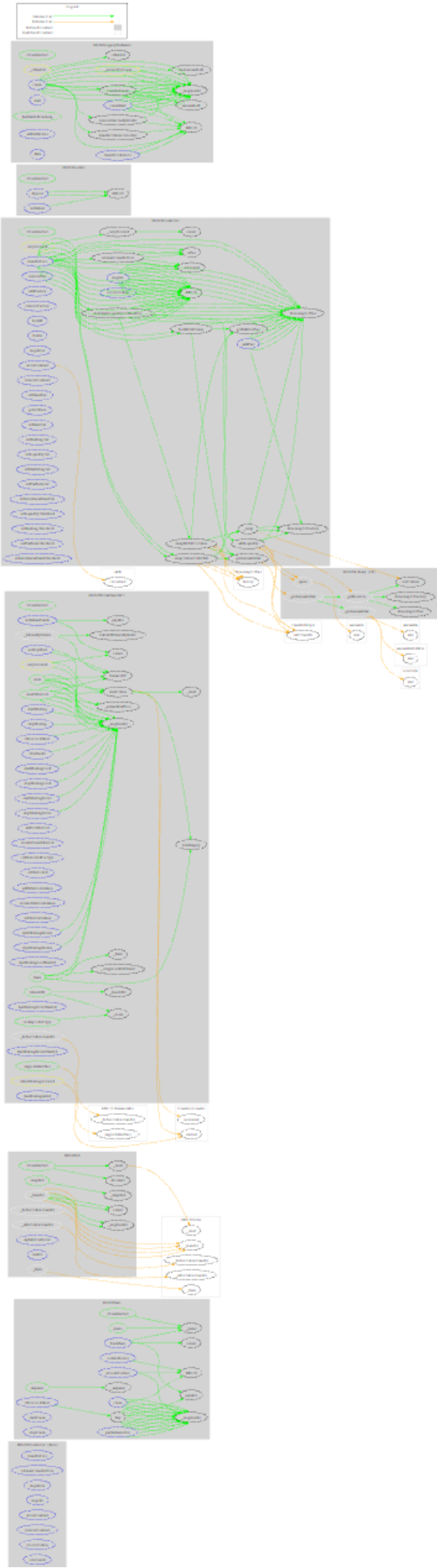
Scope of Work/Verify Claims

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. Is contract an upgradeable
2. Correct implementation of Token standard
3. Deployer cannot mint any new tokens
4. Deployer cannot burn or lock user funds
5. Deployer cannot pause the contract
6. Deployer can set fees
7. Deployer can blacklist/antisnipe address
8. Overall checkup (Smart Contract Security)

Is contract an upgradeable

Name	
Is contract an upgradeable?	**No**

Correct implementation of Token standard

ERC20				
Function	**Description**	**Exist**	**Tested**	**Verified**
totalSupply	Provides information about the total token supply			
balanceOf	Provides account balance of the owner's account			
transfer	Executes transfers of a specified number of tokens to a specified address			
transferFrom	Executes transfers of a specified number of tokens from a specified address			
approve	Allow a spender to withdraw a set number of tokens from a specified account			
allowance	Returns a set number of tokens from a spender to the owner			

ERC721				
Function	**Description**	**Exist**	**Tested**	**Verified**
BalanceOf	Count all NFTs assigned to an owner			
OwnerOf	Find the owner of an NFT			
SafeTransferFrom	Transfers the ownership of an NFT from one address to another address			
SafeTransferFrom	See above - Difference is that this function has an extra data parameter			
TransferFrom	Transfer ownership of an NFT			
Approve	Change or reaffirm the approved address for an NFT			
SetApprovalForAll	Enable or disable approval for a third party ("operator") to manage all of `msg.sender`'s assets			
GetApproved	Get the approved address for a single NFT			
IsApprovedForAll	Query if an address is an authorized operator for another address			

SupportsInterface	Query if a contract implements an interface			
Name	Provides information about the name			
Symbol	Provides information about the symbol			
TokenURI	Provides information about the TokenUri			

Write functions of contracts
v1.0

DGNX

- ♦ <Constructor>
- ♦ updateController
  - Ⓜ onlyOwner
  - Ⓜ nonReentrant
- ♦ enable
  - Ⓜ onlyOwner
- ♦ snapshot
  - Ⓜ onlyOwner

DGNXLegacyDisburser

- ♦ <Constructor>
- ♦ claimStart
  - Ⓜ _isStarted
  - Ⓜ _allowedToClaim
- ♦ claim
  - Ⓜ _isStarted
  - Ⓜ _allowedToClaim
- ♦ start
  - Ⓜ onlyOwner
- ♦ transferTokensTo
  - Ⓜ onlyOwner
- ♦ addAddresses
  - Ⓜ onlyOwner

DGNXController

- ♦ <Constructor>
- ♦ transferFees
  - Ⓜ onlyAllowed
- ♦ distributeLiquidi...
  - Ⓜ onlyAllowed
  - Ⓜ nonReentrant
- ♦ addPair
  - Ⓜ onlyAllowed
- ♦ removePair
  - Ⓜ onlyOwner
- ♦ addFactory
  - Ⓜ onlyOwner
- ♦ removeFactory
  - Ⓜ onlyOwner
- ♦ feeOff
  - Ⓜ onlyAllowed
- ♦ feeOn
  - Ⓜ onlyAllowed
- ♦ migrate
  - Ⓜ onlyAllowed
  - Ⓜ nonReentrant
- ♦ migration
  - Ⓜ onlyAllowed
  - Ⓜ nonReentrant

- ♦ recoverToken
  - Ⓜ onlyAllowed
  - Ⓜ nonReentrant
- ♦ allowContract
  - Ⓜ onlyAllowed
  - Ⓜ nonReentrant
- ♦ removeContract
  - Ⓜ onlyAllowed
- ♦ setMainPair
  - Ⓜ onlyOwner
- ♦ setBurnTax
  - Ⓜ onlyOwner
- ♦ setBackingTax
  - Ⓜ onlyOwner
- ♦ setLiquidityTax
  - Ⓜ onlyOwner
- ♦ setMarketingTax
  - Ⓜ onlyOwner
- ♦ setPlatformTax
  - Ⓜ onlyOwner
- ♦ setInvestmentFu...
  - Ⓜ onlyOwner

- ♦ setLiquidityThreshold
  - Ⓜ onlyOwner
- ♦ setBackingThreshold
  - Ⓜ onlyOwner
- ♦ setPlatformThreshold
  - Ⓜ onlyOwner
- ♦ setInvestmentFundThreshold
  - Ⓜ onlyOwner

DGNXLocker

- ♦ <Constructor>
- ♦ deposit
  - Ⓜ onlyOwner
- ♦ withdraw
  - Ⓜ onlyOwner

DGNXPrivateSaleNFT

- ⬥ <Constructor> 💰
- ⬥ withdrawFunds
- Ⓜ onlyOwner
- ⬥ airdropMint
- Ⓜ onlyOwner
- ⬥ mint 💰
- Ⓜ whenMintingAllowed
- Ⓜ nonReentrant
- ⬥ mintWhitelist 💰
- Ⓜ whenMintingAllowed
- Ⓜ nonReentrant
- ⬥ burn
- Ⓜ nonReentrant
- ⬥ addToWhitelist
- Ⓜ onlyAllowed
- ⬥ revokeFromWhitelist
- Ⓜ onlyAllowed
- ⬥ addWhitelistAdmin
- Ⓜ onlyOwner
- ⬥ revokeWhitelistAdmin
- Ⓜ onlyOwner
- ⬥ startMintingGold
- Ⓜ onlyOwner
- ⬥ stopMintingGold
- Ⓜ onlyOwner
- ⬥ startMintingSilver
- Ⓜ onlyOwner
- ⬥ stopMintingSilver
- Ⓜ onlyOwner

- ⬥ startMintingBronze
- Ⓜ onlyOwner
- ⬥ stopMintingBronze
- Ⓜ onlyOwner
- ⬥ startMinting
- Ⓜ onlyOwner
- ⬥ stopMinting
- Ⓜ onlyOwner

DGNXSale

- ⬥ finishSale
- Ⓜ onlyOwner
- ⬥ allocateForSale
- Ⓜ onlyOwner
- ⬥ lockLeftovers
- Ⓜ onlyOwner
- ⬥ pause
- Ⓜ onlyOwner
- ⬥ unpause
- Ⓜ onlyOwner
- ⬥ payEntranceFee
- Ⓜ whenNotPaused
- Ⓜ nonReentrant
- ⬥ buy 💰
- Ⓜ whenNotPaused
- Ⓜ nonReentrant
- ⬥ claim
- Ⓜ whenPaused
- Ⓜ nonReentrant
- ⬥ startClaim
- Ⓜ onlyOwner
- ⬥ stopClaim
- Ⓜ onlyOwner

Deployer cannot mint any new tokens

Name	Exist	Tested	Status
Deployer cannot mint			
Max / Total Supply	21.000.000		

Comments:

- The supply will be distributed to the owner's account at the time of deployment.

- The owner can start/stop minting of the tokens.

- Users can mint tokens when the minting will be allowed by the owner

Deployer cannot burn or lock user funds

Name	Exist	Tested	Status
Deployer cannot lock	🟩	🟩	🟥
Deployer cannot burn	🟩	🟩	🟩

Comments:

- The users can burn their own tokens.
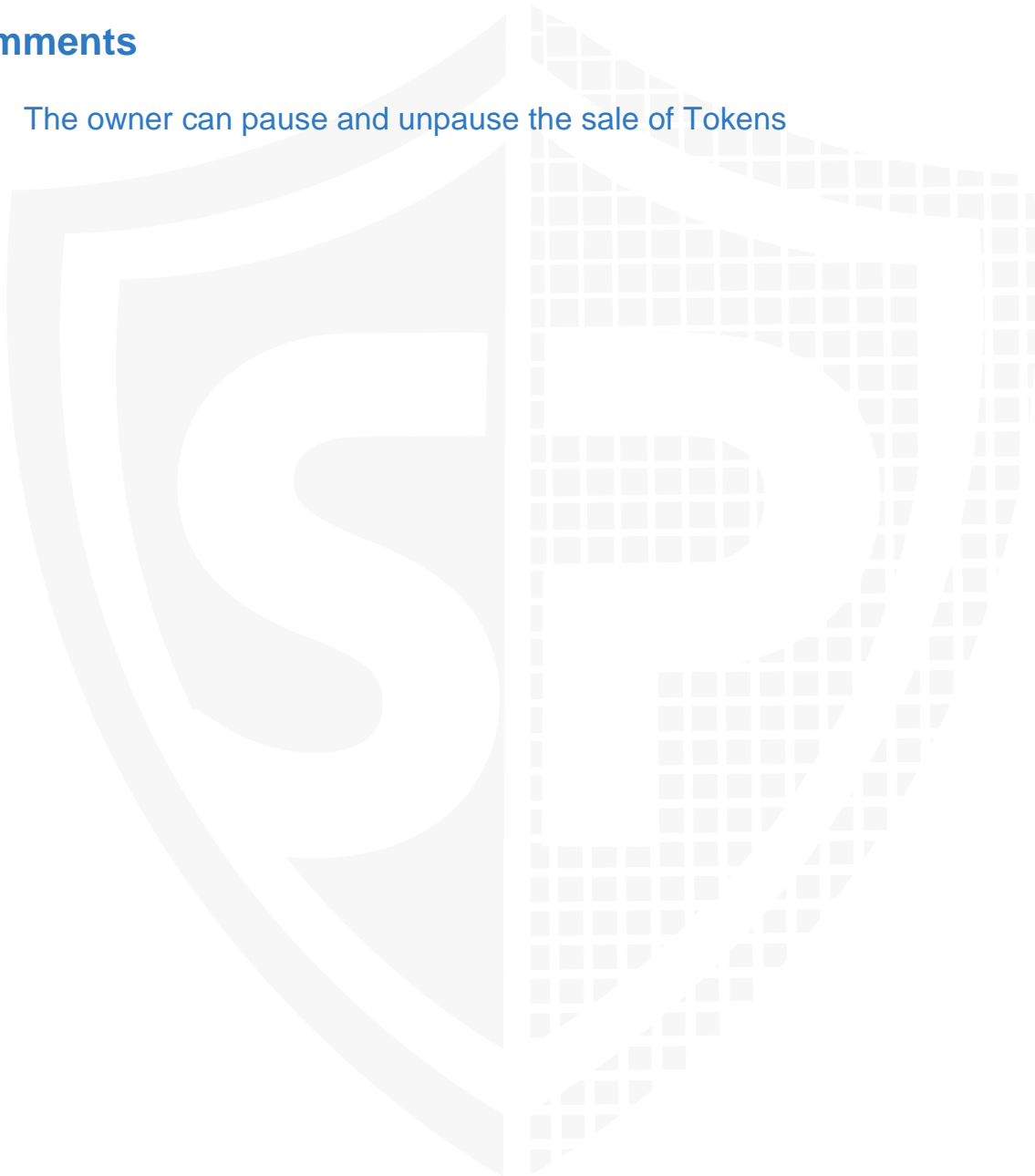
- Deployer can lock funds in the Locker contract because both deposits and withdrawals are in control of the deployer.

- Deployer can lock by setting the claimActive to false.

Deployer cannot pause the contract

Name	Exist	Tested	Status
Deployer cannot pause	🟩	🟩	🟥

Comments

The owner can pause and unpause the sale of Tokens

Deployer can set fees

Name	Exist	Tested	Status
Deployer can set fees over 25%	🟩	🟩	🟥
Deployer can set fees to nearly 100% or more	🟩	🟩	🟥

Comments:

- The fees can be set to 100% or more by the owner because there is no protection against it.

- The transferFees function will be reverted if the fees is more than 100%

Deployer cannot blacklist/antisnipe addresses

Name	Exist	Tested	Status
Deployer can blacklist/antisnipe addresses			

Comments:

- The owner can add/remove users from the whitelist.

Overall checkup (Smart Contract Security)

Tested	Verified

Legend

Attribute	Symbol
Verified / Checked	
Partly Verified	
Unverified / Not checked	
Not available	

Modifiers and public functions
v1.0

DGNX

- ♦ <Constructor>
- ♦ updateController
- Ⓜ onlyOwner
- Ⓜ nonReentrant
- ♦ enable
- Ⓜ onlyOwner
- ♦ snapshot
- Ⓜ onlyOwner

DGNXController

- ♦ <Constructor>
- ♦ transferFees
- Ⓜ onlyAllowed
- ♦ distributeLiquidi...
- Ⓜ onlyAllowed
- Ⓜ nonReentrant
- ♦ addPair
- Ⓜ onlyAllowed
- ♦ removePair
- Ⓜ onlyOwner
- ♦ addFactory
- Ⓜ onlyOwner
- ♦ removeFactory
- Ⓜ onlyOwner
- ♦ feeOff
- Ⓜ onlyAllowed
- ♦ feeOn
- Ⓜ onlyAllowed
- ♦ migrate
- Ⓜ onlyAllowed
- Ⓜ nonReentrant
- ♦ migration
- Ⓜ onlyAllowed
- Ⓜ nonReentrant

- ♦ recoverToken
- Ⓜ onlyAllowed
- Ⓜ nonReentrant
- ♦ allowContract
- Ⓜ onlyAllowed
- Ⓜ nonReentrant
- ♦ removeContract
- Ⓜ onlyAllowed
- ♦ setMainPair
- Ⓜ onlyOwner
- ♦ setBurnTax
- Ⓜ onlyOwner
- ♦ setBackingTax
- Ⓜ onlyOwner
- ♦ setLiquidityTax
- Ⓜ onlyOwner
- ♦ setMarketingTax
- Ⓜ onlyOwner
- ♦ setPlatformTax
- Ⓜ onlyOwner
- ♦ setInvestmentFu...
- Ⓜ onlyOwner

- ♦ setLiquidityThreshold
- Ⓜ onlyOwner
- ♦ setBackingThreshold
- Ⓜ onlyOwner
- ♦ setPlatformThreshold
- Ⓜ onlyOwner
- ♦ setInvestmentFundThreshold
- Ⓜ onlyOwner

DGNXLegacyDisburser

- ♦ <Constructor>
- ♦ claimStart
- Ⓜ _isStarted
- Ⓜ _allowedToClaim
- ♦ claim
- Ⓜ _isStarted
- Ⓜ _allowedToClaim
- ♦ start
- Ⓜ onlyOwner
- ♦ transferTokensTo
- Ⓜ onlyOwner
- ♦ addAddresses
- Ⓜ onlyOwner

DGNXLocker

- ♦ <Constructor>
- ♦ deposit
- Ⓜ onlyOwner
- ♦ withdraw
- Ⓜ onlyOwner

DGNXPrivateSaleNFT

- ⬥ <Constructor> 💰
- ⬥ withdrawFunds
- Ⓜ onlyOwner
- ⬥ airdropMint
- Ⓜ onlyOwner
- ⬥ mint 💰
- Ⓜ whenMintingAllowed
- Ⓜ nonReentrant
- ⬥ mintWhitelist 💰
- Ⓜ whenMintingAllowed
- Ⓜ nonReentrant
- ⬥ burn
- Ⓜ nonReentrant
- ⬥ addToWhitelist
- Ⓜ onlyAllowed
- ⬥ revokeFromWhitelist
- Ⓜ onlyAllowed
- ⬥ addWhitelistAdmin
- Ⓜ onlyOwner
- ⬥ revokeWhitelistAdmin
- Ⓜ onlyOwner
- ⬥ startMintingGold
- Ⓜ onlyOwner
- ⬥ stopMintingGold
- Ⓜ onlyOwner
- ⬥ startMintingSilver
- Ⓜ onlyOwner
- ⬥ stopMintingSilver
- Ⓜ onlyOwner

- ⬥ startMintingBronze
- Ⓜ onlyOwner
- ⬥ stopMintingBronze
- Ⓜ onlyOwner
- ⬥ startMinting
- Ⓜ onlyOwner
- ⬥ stopMinting
- Ⓜ onlyOwner

DGNXSale

- ⬥ finishSale
- Ⓜ onlyOwner
- ⬥ allocateForSale
- Ⓜ onlyOwner
- ⬥ lockLeftovers
- Ⓜ onlyOwner
- ⬥ pause
- Ⓜ onlyOwner
- ⬥ unpause
- Ⓜ onlyOwner
- ⬥ payEntranceFee
- Ⓜ whenNotPaused
- Ⓜ nonReentrant
- ⬥ buy 💰
- Ⓜ whenNotPaused
- Ⓜ nonReentrant
- ⬥ claim
- Ⓜ whenPaused
- Ⓜ nonReentrant
- ⬥ startClaim
- Ⓜ onlyOwner
- ⬥ stopClaim
- Ⓜ onlyOwner

Comments:

- The owner can set tax, modify tax and include/exclude accounts from the whitelist
- The owner will decide the bronze/silver/gold tiers for the users during the NFT private sale.
- The owner can start/stop minting of NFTs
- The owner can set the fees 100% or higher.
- Owner is able to drain the own token from contract.
- Owner can lock user funds in the DGNXSale.sol contract by setting the claimActive variable to false
- Owner can lock the buy function by setting the supply to 0.
- The deployer can change the controller contract anytime by calling the migrate function and transfer all tokens from controller contract to another contract
- There are multiple authorities in DGNXController.sol and some contracts that can make critical changes in the contract's parameters even after the ownership is renounced. Thus, ownership can never be completely renounced from such contracts.

Source Units in Scope

v1.0

File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score
contracts/interfaces/IDGNXPrivateSaleNFT.sol	———	1	11	8	4	1	7
contracts/interfaces/IDGNXController.sol	———	1	47	6	3	1	25
contracts/dgnx/DGNXSale.sol	1	———	167	163	131	14	132
contracts/dgnx/DGNX.sol	1	———	110	88	74	4	61
contracts/dgnx/DGNXPrivateSaleNFT.sol	1	———	499	456	373	31	245
contracts/dgnx/DGNXLibrary.sol	1	———	88	68	55	6	39
contracts/dgnx/DGNXLocker.sol	1	———	40	36	28	1	24
contracts/dgnx/DGNXController.sol	1	———	707	637	539	24	460
contracts/dgnx/DGNXLegacyDisburser.sol	1	———	320	298	259	7	160
Totals	**7**	**2**	**1989**	**1760**	**1466**	**89**	**1153**

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

<div style="background-color:#66ee33; text-align:center;">

AUDIT PASSED

</div>

Critical issues

<div style="background-color:#66ee33; text-align:center; color:green;">
No critical issues
</div>

High issues

<div style="background-color:#66ee33; text-align:center; color:green;">
No high issues
</div>

Medium issues

Issue	File	Type	Line	Description
#1	DGNXController.sol	Tax can be 100%	664-684	The owner can set the tax as 100% or more which may result in loss and lock of user funds.

Low issues

Issue	File	Type	Line	Description
#1	DGNXController.sol	Missing Events	548,552,609, 624,640,645, 654,664-703	Emit an event for critical parameter changes.
#2	DGNXController.sol	Missing zero check	640	Check that the address is not zero
#3	DGNXGovernor.sol	Shadowing Local Variables	19	Rename the local variables that shadow another component
#4	PrivateSaleNFT.sol	Shadowing Local Variables	84,85	Rename the local variables that shadow another component
#5	DGNXLegacyDisburser.sol	Missing Events	43,216,247	Emit an event for critical parameter changes.
#6	DGNXSale.sol	Missing Events	157,164	Emit an event for critical parameter changes.

#7	PrivateSaleNFT.sol	Drain contract tokens	187	Owner is able to drain the own token from contract. We recommend to prevent passing the own contract address
#8	DGNXLegacyDisburser.sol	Drain contract tokens	239,49	Owner is able to drain the own token from contract. We recommend to prevent passing the own contract address
#9	All	Floating Pragma	-	The current pragma Solidity directive is "^0.8.4". Contracts should be deployed with the same compiler version and flags that they have been tested with thoroughly. Locking the pragma helps to ensure that contracts do not accidentally get deployed using other versions.

Informational issues

Issue	File	Type	Line	Description
#1	All	NatSpec documentation missing	—	If you started to comment your code, also comment all other functions, variables etc.
#2	DGNXController.sol	Unused State Variable	89	Make sure that the spelling are correct to improve user readability.
#3	DGNXController.sol	Missing Inheritance	18	Contract should inherit from IDGNXController
#4	DGNXLegacyDisburser.sol	Uninitialized Local Variable	269,270	Make sure that all the variables are initialized.
#5	DGNXPrivateSaleNFT.sol	Dead Code	170	This function was never called in the contract and should be removed.

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.

23. August, 2022:

- There is still an owner (Owner still has not renounced ownership)
- For DGNXLegacyDisburser.sol we recommend using chainlink VRF for randomization.
- Read the 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	**NOT 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-	Presence of unused variables	CWE-1164: Irrelevant Code	**NOT PASSED**

1 3 1			
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-	Write to Arbitrary	CWE-123: Write-what-where Condition	**PASSED**

[1 2 4](#)	Storage Location		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**