Cyberscope

Audit Report

Wagon

January 2023

Type	ERC20
Network	ETH
Address	0xd50c8a17d5c4b8e2d984933C7E37e5B92d687B8D
Audited by	© cyberscope

Table of Contents

Table of Contents 1
Review 3
Audit Updates 3
Source Files 4
Analysis 6
ST - Stops Transactions 7
Description 7
Recommendation 7
OTUT - Transfers User's Tokens 8
Description 8
Recommendation 8
MT - Mints Tokens 9
Description 9
Recommendation 9
BC - Blacklists Addresses 10
Description 10
Recommendation 10
Diagnostics 11
L04 - Conformance to Solidity Naming Conventions 12
Description 12
Recommendation 12
L16 - Validate Variable Setters 14
Description 14
Recommendation 14
L19 - Stable Compiler Version 15
Description 15
Recommendation 15
Contract Functions 16
Inheritance Graph 26
Flow Graph 27
Summary 28

Disclaimer 29

About Cyberscope 30

Review

Contract Name	Wagon
Compiler Version	v0.8.9+commit.e5eed63a
Optimization	200 runs
Explorer	https://etherscan.io/address/0xd50c8a17d5c4b8e2d984933c7e37e5b92d687b8d
Address	0xd50c8a17d5c4b8e2d984933c7e37e5b92d687b8d
Network	ETH
Symbol	WAG
Decimals	18
Total Supply	100,000,000

Audit Updates

Initial Audit	09 Jan 2023 https://github.com/cyberscope-io/audits/tree/main/1-wag/v1/audit.pdf
Corrected Phase 2	13 Jan 2023

Source Files

Filename	SHA256
@openzeppelin/contracts/access/AccessControl.sol	5af1771388b4fe634e0a566716e32c6d00a5372875099127b274d4cf8a94e9d2
@openzeppelin/contracts/access/IAccessControl.sol	d03c1257f2094da6c86efa7aa09c1c07ebd33dd31046480c5097bc2542140e45
@openzeppelin/contracts/governance/utils/IVotes.sol	55fe90680900ea253e4e5b11d9b6ab5c4ff3e85e48ffb94c8b2c29694d01312b
@openzeppelin/contracts/security/Pausable.sol	2072248d2f79e661c149fd6a6593a8a3f038466557c9b75e50e0b001bcb5cf97
@openzeppelin/contracts/token/ERC20/ERC20.sol	5031430cc2613c32736d598037d3075985a2a09e61592a013dbd09a5bc2041b8
@openzeppelin/contracts/token/ERC20/extensions/draft-ERC20Permit.sol	d070a08919d4a38aa08043c687d1fe1522098b212d2e185aedf2f37275b64087
@openzeppelin/contracts/token/ERC20/extensions/draft-IERC20Permit.sol	3e7aa0e0f69eec8f097ad664d525e7b3f0a3fda8dcdd97de5433ddb131db86ef
@openzeppelin/contracts/token/ERC20/extensions/ERC20Burnable.sol	0344809a1044e11ece2401b4f7288f414ea41fa9d1dad24143c84b737c9fc02e
@openzeppelin/contracts/token/ERC20/extensions/ERC20Snapshot.sol	62560c159bc1b088a9d69b1676dfee8f25c750c583a5edf3115a7d72451c94f5
@openzeppelin/contracts/token/ERC20/extensions/ERC20Votes.sol	fb449cd9e8ce63e968e8b5c3d39e64f9928a854fcfa4db33d6a853f890e47fd6
@openzeppelin/contracts/token/ERC20/extensions/IERC20Metadata.sol	af5c8a77965cc82c33b7ff844deb9826166689e55dc037a7f2f790d057811990
@openzeppelin/contracts/token/ERC20/IERC20.sol	94f23e4af51a18c2269b355b8c7cf4db8003d075c9c541019eb8dcf4122864d5

@openzeppelin/contracts/utils/Arrays.sol	7aadd135b55a263885c171517af1fae9ac1fe6573b34e041b447c218cf5b4f64
@openzeppelin/contracts/utils/Context.sol	1458c260d010a08e4c20a4a517882259a23a4baa0b5bd9add9fb6d6a1549814a
@openzeppelin/contracts/utils/Counters.sol	2fdcb1343e5621385b62e57b5c7775607c272122b6f2dc77da8f84828aa40cd0
@openzeppelin/contracts/utils/cryptography/draft-EIP712.sol	fc0e6c5d7184bd03b8deae6ca9a48a1eaaecf9f5e4703611aabfb63401e6d43f
@openzeppelin/contracts/utils/cryptography/ECDSA.sol	4e45d53327d561848fbcf381262ec5c0ac91b2f1f06432210bf76db55279d945
@openzeppelin/contracts/utils/introspection/ERC165.sol	8806a632d7b656cadb8133ff8f2acae4405b3a64d8709d93b0fa6a216a8a6154
@openzeppelin/contracts/utils/introspection/IERC165.sol	701e025d13ec6be09ae892eb029cd83b3064325801d73654847a5fb11c58b1e5
@openzeppelin/contracts/utils/math/Math.sol	929523c09910460ad708c75878d89b9fbed12b65cb5d8b670200c793131072f4
@openzeppelin/contracts/utils/math/SafeCast.sol	e44469cf1affcd59005dc9c69df91af9c7b93e6bc4095148232f86ba9e7f749d
@openzeppelin/contracts/utils/Strings.sol	34127ad0054df5963b0fd694c1b313d17e9114a2f426b85526d6d976210298ab
contracts/Wagon.sol	cde60758aa3462fc0ace6cc1c663498b0d70df5fdede5ae7d43fe2b3f022ee2e

Analysis

● Critical ● Medium ● Minor / Informative ● Pass

Severity	Code	Description	Status
●	ST	Stops Transactions	Unresolved
●	OCTD	Transfers Contract's Tokens	Passed
●	OTUT	Transfers User's Tokens	Unresolved
●	ELFM	Exceeds Fees Limit	Passed
●	ULTW	Transfers Liquidity to Team Wallet	Passed
●	MT	Mints Tokens	Unresolved
●	BT	Burns Tokens	Passed
●	BC	Blacklists Addresses	Unresolved

ST - Stops Transactions

Criticality	Minor / Informative
Location	contracts/Wagon.sol#L77
Status	Unresolved

Description

The contract owner has the authority to stop the transactions for all users. The owner may take advantage of it by setting the `pause` function, which will pause all transfers.

```
function pause() public onlyRole(PAUSER_ROLE) {
  _pause();
}
...
function _beforeTokenTransfer(address from, address to, uint256 amount)
  internal
  whenNotPaused
  isNotBlackListed(from, to)
  override(ERC20, ERC20Snapshot)
{
  super._beforeTokenTransfer(from, to, amount);
}
```

Recommendation

The team should carefully manage the private keys of the owner's account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions. That risk can be prevented by temporarily locking the contract or renouncing ownership.

OTUT - Transfers User's Tokens

Criticality	Critical
Location	contracts/Wagon.sol#L153
Status	Unresolved

Description

The contract owner has the authority to transfer the balance of a blacklisted user's contract to the owner's contract. The owner may take advantage of it by calling the `recoverBlackFunds` function.

```
function recoverBlackFunds (address _blackListedUser) public
onlyRole(BLACKLISTER_ROLE) {
  require(isBlackListed[_blackListedUser], "User is not blacklisted");
  uint dirtyFunds = balanceOf(_blackListedUser);
  _transfer(_blackListedUser, emergencyAccount, dirtyFunds);
  emit RecoveredBlackFunds(_blackListedUser, dirtyFunds);
}
```

Recommendation

The team should carefully manage the private keys of the owner’s account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions. That risk can be prevented by temporarily locking the contract or renouncing ownership.

MT - Mints Tokens

Criticality	Critical
Location	contracts/Wagon.sol#L91
Status	Unresolved

Description

The contract owner has the authority to mint tokens. The owner may take advantage of it by calling the `mint` function. As a result, the contract tokens will be highly inflated.

```
function mint(address to, uint256 amount) public onlyRole(MINTER_ROLE) {
  _mint(to, amount);
}
```

Recommendation

The team should carefully manage the private keys of the owner's account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions. That risk can be prevented by temporarily locking the contract or renouncing ownership.

BC - Blacklists Addresses

Criticality	Medium
Location	contracts/Wagon.sol#L137
Status	Unresolved

Description

The contract owner has the authority to stop addresses from transactions. The owner may take advantage of it by calling the `addBlackList` function.

```
function addBlackList (address _evilUser) public onlyRole(BLACKLISTER_ROLE) {
  isBlackListed[_evilUser] = true;
  emit AddedBlackList(_evilUser);
}
```

Recommendation

The team should carefully manage the private keys of the owner's account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions. That risk can be prevented by temporarily locking the contract or renouncing ownership.

Diagnostics

Critical Medium Minor / Informative

Severity	Code	Description	Status
●	L04	Conformance to Solidity Naming Conventions	Unresolved
●	L16	Validate Variable Setters	Unresolved
●	L19	Stable Compiler Version	Unresolved

L04 - Conformance to Solidity Naming Conventions

Criticality	Minor / Informative
Location	contracts/Wagon.sol#L58,130,137,145,153
Status	Unresolved

Description

The Solidity style guide is a set of guidelines for writing clean and consistent Solidity code. Adhering to a style guide can help improve the readability and maintainability of the Solidity code, making it easier for others to understand and work with.

The followings are a few key points from the Solidity style guide:

1. Use camelCase for function and variable names, with the first letter in lowercase (e.g., myVariable, updateCounter).
2. Use PascalCase for contract, struct, and enum names, with the first letter in uppercase (e.g., MyContract, UserStruct, ErrorEnum).
3. Use uppercase for constant variables and enums (e.g., MAX_VALUE, ERROR_CODE).
4. Use indentation to improve readability and structure.
5. Use spaces between operators and after commas.
6. Use comments to explain the purpose and behavior of the code.
7. Keep lines short (around 120 characters) to improve readability.

```
getCurrentSnapshotId();

n removeBlackL
/ Recover WAG tok
irtyFunds = balanceO

 uint256 amount)
```

Recommendation

By following the Solidity naming convention guidelines, the codebase increased the readability, maintainability, and makes it easier to work with.

Find more information on the Solidity documentation https://docs.soliditylang.org/en/v0.8.17/style-guide.html#naming-convention.

L16 - Validate Variable Setters

Criticality	Minor / Informative
Location	contracts/Wagon.sol#L59
Status	Unresolved

Description

The contract performs operations on variables that have been configured on user-supplied input. These variables are missing of proper check for the case where a value is zero. This can lead to problems when the contract is executed, as certain actions may not be properly handled when the value is zero.

```
emergencyAccount = _newEmergencyAddress;
```

Recommendation

By adding the proper check, the contract will not allow the variables to be configured with zero value. This will ensure that the contract can handle all possible input values and avoid unexpected behavior or errors. Hence, it can help to prevent the contract from being exploited or operating unexpectedly.

L19 - Stable Compiler Version

Criticality	Minor / Informative
Location	contracts/Wagon.sol#L2
Status	Unresolved

Description

The `^` symbol indicates that any version of Solidity that is compatible with the specified version (i.e., any version that is a higher minor or patch version) can be used to compile the contract. The version lock is a mechanism that allows the author to specify a minimum version of the Solidity compiler that must be used to compile the contract code. This is useful because it ensures that the contract will be compiled using a version of the compiler that is known to be compatible with the code.

```
pragma solidity ^0.8.9;
```

Recommendation

The team is advised to lock the pragma to ensure the stability of the codebase. The locked pragma version ensures that the contract will not be deployed with an unexpected version. An unexpected version may produce vulnerabilities and undiscovered bugs. The compiler should be configured to the lowest version that provides all the required functionality for the codebase. As a result, the project will be compiled in a well-tested LTS (Long Term Support) environment.

Contract Functions

Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
AccessControl	Implementation	Context, IAccessControl, ERC165		
	supportsInterface	Public		-
	hasRole	Public		-
	_checkRole	Internal		
	_checkRole	Internal		
	getRoleAdmin	Public		-
	grantRole	Public	✓	onlyRole
	revokeRole	Public	✓	onlyRole
	renounceRole	Public	✓	-
	_setupRole	Internal	✓	
	_setRoleAdmin	Internal	✓	
	_grantRole	Internal	✓	
	_revokeRole	Internal	✓	
IAccessControl	Interface			
	hasRole	External		-
	getRoleAdmin	External		-
	grantRole	External	✓	-
	revokeRole	External	✓	-
	renounceRole	External	✓	-
IVotes	Interface			

	getVotes	External		-
	getPastVotes	External		-
	getPastTotalSupply	External		-
	delegates	External		-
	delegate	External	✓	-
	delegateBySig	External	✓	-
Pausable	Implementation	Context		
		Public	✓	-
	paused	Public		-
	_requireNotPaused	Internal		
	_requirePaused	Internal		
	_pause	Internal	✓	whenNotPaused
	_unpause	Internal	✓	whenPaused
ERC20	Implementation	Context, IERC20, IERC20Metadata		
		Public	✓	-
	name	Public		-
	symbol	Public		-
	decimals	Public		-
	totalSupply	Public		-
	balanceOf	Public		-
	transfer	Public	✓	-
	allowance	Public		-
	approve	Public	✓	-
	transferFrom	Public	✓	-
	increaseAllowance	Public	✓	-
	decreaseAllowance	Public	✓	-

	_transfer	Internal	✓	
	_mint	Internal	✓	
	_burn	Internal	✓	
	_approve	Internal	✓	
	_spendAllowance	Internal	✓	
	_beforeTokenTransfer	Internal	✓	
	_afterTokenTransfer	Internal	✓	
ERC20Permit	Implementation	ERC20, IERC20Permit, EIP712		
		Public	✓	EIP712
	permit	Public	✓	-
	nonces	Public		-
	DOMAIN_SEPARATOR	External		-
	_useNonce	Internal	✓	
IERC20Permit	Interface			
	permit	External	✓	-
	nonces	External		-
	DOMAIN_SEPARATOR	External		-
ERC20Burnable	Implementation	Context, ERC20		
	burn	Public	✓	-
	burnFrom	Public	✓	-
ERC20Snapshot	Implementation	ERC20		
	_snapshot	Internal	✓	
	_getCurrentSnapshotId	Internal		
	balanceOfAt	Public		-

	totalSupplyAt	Public		-
	_beforeTokenTransfer	Internal	✓	
	_valueAt	Private		
	_updateAccountSnapshot	Private	✓	
	_updateTotalSupplySnapshot	Private	✓	
	_updateSnapshot	Private	✓	
	_lastSnapshotId	Private		
ERC20Votes	Implementation	IVotes, ERC20Permit		
	checkpoints	Public		-
	numCheckpoints	Public		-
	delegates	Public		-
	getVotes	Public		-
	getPastVotes	Public		-
	getPastTotalSupply	Public		-
	_checkpointsLookup	Private		
	delegate	Public	✓	-
	delegateBySig	Public	✓	-
	_maxSupply	Internal		
	_mint	Internal	✓	
	_burn	Internal	✓	
	_afterTokenTransfer	Internal	✓	
	_delegate	Internal	✓	
	_moveVotingPower	Private	✓	
	_writeCheckpoint	Private	✓	
	_add	Private		
	_subtract	Private		

IERC20Metadata	Interface	IERC20		
	name	External		-
	symbol	External		-
	decimals	External		-
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
Arrays	Library			
	findUpperBound	Internal		
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		
Counters	Library			
	current	Internal		
	increment	Internal	✓	
	decrement	Internal	✓	
	reset	Internal	✓	
EIP712	Implementation			
		Public	✓	-
	_domainSeparatorV4	Internal		

	_buildDomainSeparator	Private		
	_hashTypedDataV4	Internal		
ECDSA	Library			
	_throwError	Private		
	tryRecover	Internal		
	recover	Internal		
	tryRecover	Internal		
	recover	Internal		
	tryRecover	Internal		
	recover	Internal		
	toEthSignedMessageHash	Internal		
	toEthSignedMessageHash	Internal		
	toTypedDataHash	Internal		
ERC165	Implementation	IERC165		
	supportsInterface	Public		-
IERC165	Interface			
	supportsInterface	External		-
Math	Library			
	max	Internal		
	min	Internal		
	average	Internal		
	ceilDiv	Internal		
	mulDiv	Internal		
	mulDiv	Internal		
	sqrt	Internal		

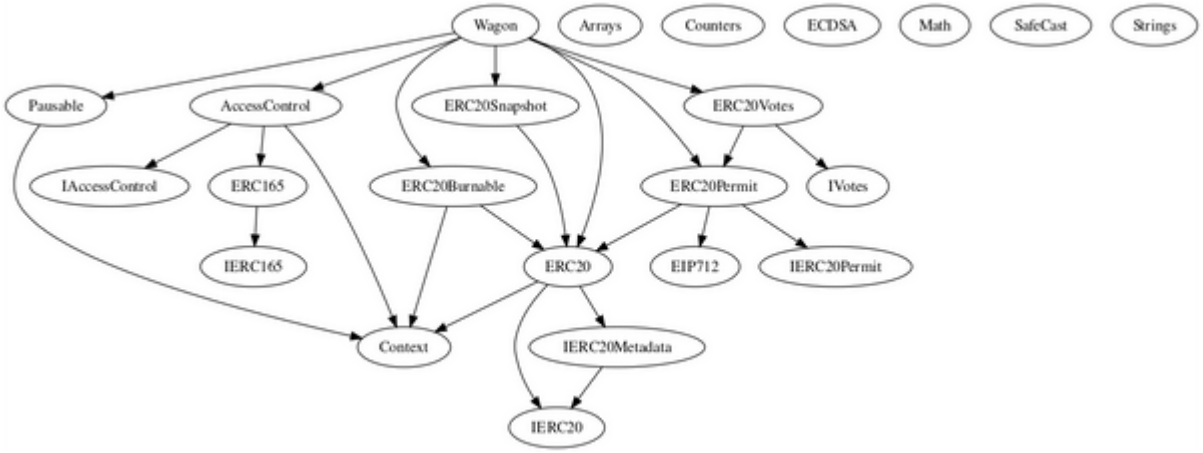
	sqrt	Internal		
SafeCast	Library			
	toUint248	Internal		
	toUint240	Internal		
	toUint232	Internal		
	toUint224	Internal		
	toUint216	Internal		
	toUint208	Internal		
	toUint200	Internal		
	toUint192	Internal		
	toUint184	Internal		
	toUint176	Internal		
	toUint168	Internal		
	toUint160	Internal		
	toUint152	Internal		
	toUint144	Internal		
	toUint136	Internal		
	toUint128	Internal		
	toUint120	Internal		
	toUint112	Internal		
	toUint104	Internal		
	toUint96	Internal		
	toUint88	Internal		
	toUint80	Internal		
	toUint72	Internal		
	toUint64	Internal		
	toUint56	Internal		
	toUint48	Internal		

	toUint40	Internal		
	toUint32	Internal		
	toUint24	Internal		
	toUint16	Internal		
	toUint8	Internal		
	toUint256	Internal		
	toInt248	Internal		
	toInt240	Internal		
	toInt232	Internal		
	toInt224	Internal		
	toInt216	Internal		
	toInt208	Internal		
	toInt200	Internal		
	toInt192	Internal		
	toInt184	Internal		
	toInt176	Internal		
	toInt168	Internal		
	toInt160	Internal		
	toInt152	Internal		
	toInt144	Internal		
	toInt136	Internal		
	toInt128	Internal		
	toInt120	Internal		
	toInt112	Internal		
	toInt104	Internal		
	toInt96	Internal		
	toInt88	Internal		
	toInt80	Internal		
	toInt72	Internal		

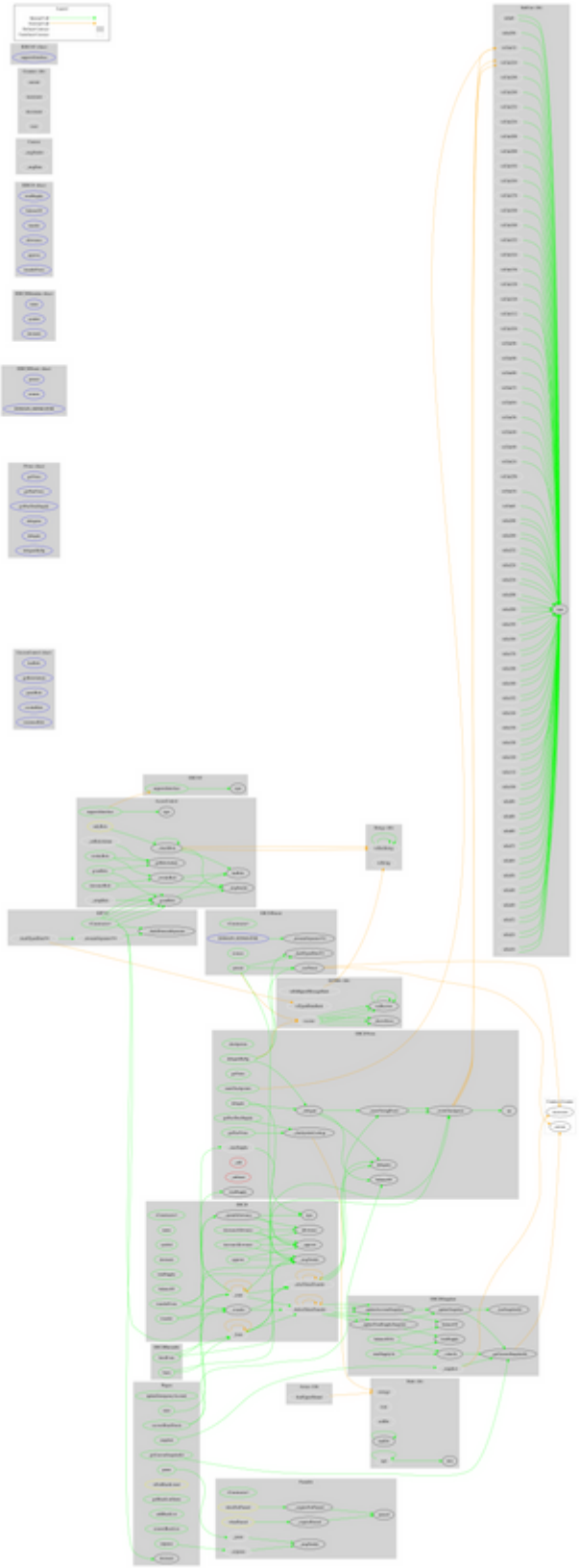
	toInt64	Internal		
	toInt56	Internal		
	toInt48	Internal		
	toInt40	Internal		
	toInt32	Internal		
	toInt24	Internal		
	toInt16	Internal		
	toInt8	Internal		
	toInt256	Internal		
Strings	Library			
	toString	Internal		
	toHexString	Internal		
	toHexString	Internal		
	toHexString	Internal		
Wagon	Implementation	ERC20, ERC20Burnable, ERC20Snapshot, AccessControl, Pausable, ERC20Permit, ERC20Votes		
		Public	✓	ERC20 ERC20Permit
	updateEmergencyAccount	Public	✓	onlyRole
	snapshot	Public	✓	onlyRole
	getCurrentSnapshotId	Public		-
	pause	Public	✓	onlyRole
	unpause	Public	✓	onlyRole

	mint	Public	✓	onlyRole
	_beforeTokenTransfer	Internal	✓	whenNotPaused isNotBlackListed
	getBlackListStatus	Public		-
	addBlackList	Public	✓	onlyRole
	removeBlackList	Public	✓	onlyRole
	recoverBlackFunds	Public	✓	onlyRole
	_afterTokenTransfer	Internal	✓	
	_mint	Internal	✓	
	_burn	Internal	✓	

Inheritance Graph



Flow Graph



Summary

There are some functions that can be abused by the owner like stop transactions, transfer the user's tokens, mint tokens and blacklist addresses. if the contract owner abuses the mint functionality, then the contract will be highly inflated. A multi-wallet signing pattern will provide security against potential hacks. Temporarily locking the contract or renouncing ownership will eliminate all the contract threats.

Disclaimer

The information provided in this report does not constitute investment, financial or trading advice and you should not treat any of the document's content as such. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes nor may copies be delivered to any other person other than the Company without Cyberscope’s prior written consent. This report is not nor should be considered an “endorsement” or “disapproval” of any particular project or team. This report is not nor should be regarded as an indication of the economics or value of any “product” or “asset” created by any team or project that contracts Cyberscope to perform a security assessment. This document does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors' business, business model or legal compliance. This report should not be used in any way to make decisions around investment or involvement with any particular project. This report represents an extensive assessment 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 Cyberscope’s position is that each company and individual are responsible for their own due diligence and continuous security Cyberscope’s goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies and in no way claims any guarantee of security or functionality of the technology we agree to analyze. The assessment services provided by Cyberscope are subject to dependencies and are under continuing development. You agree that your access and/or use including but not limited to any services reports and materials will be at your sole risk on an as-is where-is and as-available basis Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives false negatives and other unpredictable results. The services may access and depend upon multiple layers of third parties.

About Cyberscope

Cyberscope is a blockchain cybersecurity company that was founded with the vision to make web3.0 a safer place for investors and developers. Since its launch, it has worked with thousands of projects and is estimated to have secured tens of millions of investors' funds.

Cyberscope is one of the leading smart contract audit firms in the crypto space and has built a high-profile network of clients and partners.



The Cyberscope team

https://www.cyberscope.io

Cyberscope

Audit Report

Wagon

January 2023

SHA256 393a32169db36ac97d901b5c2a4441c67d08cadda2a531976633fd128d2a1c89

Audited by © cyberscope

Table of Contents

Table of Contents 1
Review 3
Audit Updates 3
Source Files 4
Analysis 6
ST - Stops Transactions 7
Description 7
Recommendation 7
OTUT - Transfers User's Tokens 8
Description 8
Recommendation 8
MT - Mints Tokens 9
Description 9
Recommendation 9
BC - Blacklists Addresses 10
Description 10
Recommendation 10
Diagnostics 11
L04 - Conformance to Solidity Naming Conventions 12
Description 12
Recommendation 12
L16 - Validate Variable Setters 14
Description 14
Recommendation 14
L19 - Stable Compiler Version 15
Description 15
Recommendation 15
Contract Functions 16
Inheritance Graph 26
Flow Graph 27
Summary 28

Disclaimer 29

About Cyberscope 30

Review

Contract Name	Wagon
Compiler Version	v0.8.9+commit.e5eed63a
Optimization	200 runs
Testing Deploy	https://testnet.bscscan.com/address/0x8f9fdc44a2092c33bd6ad4caa135c54d2d654740
Address	0x8f9fdc44a2092c33bd6ad4caa135c54d2d654740
Network	BSC_TESTNET
Symbol	WAG
Decimals	18
Total Supply	1,000,000,000

Audit Updates

Initial Audit	09 Jan 2023

Source Files

Filename	SHA256
@openzeppelin/contracts/access/AccessControl.sol	5af1771388b4fe634e0a566716e32c6d00a5372875099127b274d4cf8a94e9d2
@openzeppelin/contracts/access/IAccessControl.sol	d03c1257f2094da6c86efa7aa09c1c07ebd33dd31046480c5097bc2542140e45
@openzeppelin/contracts/governance/utils/IVotes.sol	55fe90680900ea253e4e5b11d9b6ab5c4ff3e85e48ffb94c8b2c29694d01312b
@openzeppelin/contracts/security/Pausable.sol	2072248d2f79e661c149fd6a6593a8a3f038466557c9b75e50e0b001bcb5cf97
@openzeppelin/contracts/token/ERC20/ERC20.sol	5031430cc2613c32736d598037d3075985a2a09e61592a013dbd09a5bc2041b8
@openzeppelin/contracts/token/ERC20/extensions/draft-ERC20Permit.sol	d070a08919d4a38aa08043c687d1fe1522098b212d2e185aedf2f37275b64087
@openzeppelin/contracts/token/ERC20/extensions/draft-IERC20Permit.sol	3e7aa0e0f69eec8f097ad664d525e7b3f0a3fda8dcdd97de5433ddb131db86ef
@openzeppelin/contracts/token/ERC20/extensions/ERC20Burnable.sol	0344809a1044e11ece2401b4f7288f414ea41fa9d1dad24143c84b737c9fc02e
@openzeppelin/contracts/token/ERC20/extensions/ERC20Snapshot.sol	62560c159bc1b088a9d69b1676dfee8f25c750c583a5edf3115a7d72451c94f5
@openzeppelin/contracts/token/ERC20/extensions/ERC20Votes.sol	fb449cd9e8ce63e968e8b5c3d39e64f9928a854fcfa4db33d6a853f890e47fd6
@openzeppelin/contracts/token/ERC20/extensions/IERC20Metadata.sol	af5c8a77965cc82c33b7ff844deb9826166689e55dc037a7f2f790d057811990
@openzeppelin/contracts/token/ERC20/IERC20.sol	94f23e4af51a18c2269b355b8c7cf4db8003d075c9c541019eb8dcf4122864d5
@openzeppelin/contracts/utils/Arrays.sol	7aadd135b55a263885c171517af1fae9ac1fe6573b34e041b447c218cf5b4f64

@openzeppelin/contracts/utils/Context.sol	1458c260d010a08e4c20a4a517882259a23a4baa0b5bd9add9fb6d6a1549814a
@openzeppelin/contracts/utils/Counters.sol	2fdcb1343e5621385b62e57b5c7775607c272122b6f2dc77da8f84828aa40cd0
@openzeppelin/contracts/utils/cryptography/draft-EIP712.sol	fc0e6c5d7184bd03b8deae6ca9a48a1eaaecf9f5e4703611aabfb63401e6d43f
@openzeppelin/contracts/utils/cryptography/ECDSA.sol	4e45d53327d561848fbcf381262ec5c0ac91b2f1f06432210bf76db55279d945
@openzeppelin/contracts/utils/introspection/ERC165.sol	8806a632d7b656cadb8133ff8f2acae4405b3a64d8709d93b0fa6a216a8a6154
@openzeppelin/contracts/utils/introspection/IERC165.sol	701e025d13ec6be09ae892eb029cd83b3064325801d73654847a5fb11c58b1e5
@openzeppelin/contracts/utils/math/Math.sol	929523c09910460ad708c75878d89b9fbed12b65cb5d8b670200c793131072f4
@openzeppelin/contracts/utils/math/SafeCast.sol	e44469cf1affcd59005dc9c69df91af9c7b93e6bc4095148232f86ba9e7f749d
@openzeppelin/contracts/utils/Strings.sol	34127ad0054df5963b0fd694c1b313d17e9114a2f426b85526d6d976210298ab
contracts/Wagon.sol	393a32169db36ac97d901b5c2a4441c67d08cadda2a531976633fd128d2a1c89

Analysis

● Critical ● Medium ● Minor / Informative ● Pass

Severity	Code	Description	Status
●	ST	Stops Transactions	Unresolved
●	OCTD	Transfers Contract's Tokens	Passed
●	OTUT	Transfers User's Tokens	Unresolved
●	ELFM	Exceeds Fees Limit	Passed
●	ULTW	Transfers Liquidity to Team Wallet	Passed
●	MT	Mints Tokens	Unresolved
●	BT	Burns Tokens	Passed
●	BC	Blacklists Addresses	Unresolved

ST - Stops Transactions

Criticality	Minor / Informative
Location	contracts/Wagon.sol#L77
Status	Unresolved

Description

The contract owner has the authority to stop the transactions for all users. The owner may take advantage of it by setting the `pause` function, which will pause all transfers.

```
function pause() public onlyRole(PAUSER_ROLE) {
  _pause();
}
...
function _beforeTokenTransfer(address from, address to, uint256 amount)
  internal
  whenNotPaused
  isNotBlackListed(from, to)
  override(ERC20, ERC20Snapshot)
{
  super._beforeTokenTransfer(from, to, amount);
}
```

Recommendation

The team should carefully manage the private keys of the owner's account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions. That risk can be prevented by temporarily locking the contract or renouncing ownership.

OTUT - Transfers User's Tokens

Criticality	Critical
Location	contracts/Wagon.sol#L153
Status	Unresolved

Description

The contract owner has the authority to transfer the balance of a blacklisted user's contract to the owner's contract. The owner may take advantage of it by calling the `recoverBlackFunds` function.

```
function recoverBlackFunds (address _blackListedUser) public
onlyRole(BLACKLISTER_ROLE) {
  require(isBlackListed[_blackListedUser], "User is not blacklisted");
  uint dirtyFunds = balanceOf(_blackListedUser);
  _transfer(_blackListedUser, emergencyAccount, dirtyFunds);
  emit RecoveredBlackFunds(_blackListedUser, dirtyFunds);
}
```

Recommendation

The team should carefully manage the private keys of the owner’s account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions. That risk can be prevented by temporarily locking the contract or renouncing ownership.

MT - Mints Tokens

Criticality	Critical
Location	contracts/Wagon.sol#L91
Status	Unresolved

Description

The contract owner has the authority to mint tokens. The owner may take advantage of it by calling the `mint` function. As a result, the contract tokens will be highly inflated.

```
function mint(address to, uint256 amount) public onlyRole(MINTER_ROLE) {
  _mint(to, amount);
}
```

Recommendation

The team should carefully manage the private keys of the owner's account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions. That risk can be prevented by temporarily locking the contract or renouncing ownership.

BC - Blacklists Addresses

Criticality	Medium
Location	contracts/Wagon.sol#L137
Status	Unresolved

Description

The contract owner has the authority to stop addresses from transactions. The owner may take advantage of it by calling the `addBlackList` function.

```
function addBlackList (address _evilUser) public onlyRole(BLACKLISTER_ROLE) {
  isBlackListed[_evilUser] = true;
  emit AddedBlackList(_evilUser);
}
```

Recommendation

The team should carefully manage the private keys of the owner's account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions. That risk can be prevented by temporarily locking the contract or renouncing ownership.

Diagnostics

Critical Medium Minor / Informative

Severity	Code	Description	Status
●	L04	Conformance to Solidity Naming Conventions	Unresolved
●	L16	Validate Variable Setters	Unresolved
●	L19	Stable Compiler Version	Unresolved

L04 - Conformance to Solidity Naming Conventions

Criticality	Minor / Informative
Location	contracts/Wagon.sol#L58,130,137,145,153
Status	Unresolved

Description

The Solidity style guide is a set of guidelines for writing clean and consistent Solidity code. Adhering to a style guide can help improve the readability and maintainability of the Solidity code, making it easier for others to understand and work with.

The followings are a few key points from the Solidity style guide:

1. Use camelCase for function and variable names, with the first letter in lowercase (e.g., myVariable, updateCounter).
2. Use PascalCase for contract, struct, and enum names, with the first letter in uppercase (e.g., MyContract, UserStruct, ErrorEnum).
3. Use uppercase for constant variables and enums (e.g., MAX_VALUE, ERROR_CODE).
4. Use indentation to improve readability and structure.
5. Use spaces between operators and after commas.
6. Use comments to explain the purpose and behavior of the code.
7. Keep lines short (around 120 characters) to improve readability.

```
getCurrentSnapshotId();

n removeBlackL
/ Recover WAG tok
irtyFunds = balanceO

 uint256 amount)
```

Recommendation

By following the Solidity naming convention guidelines, the codebase increased the readability, maintainability, and makes it easier to work with.

Find more information on the Solidity documentation https://docs.soliditylang.org/en/v0.8.17/style-guide.html#naming-convention.

L16 - Validate Variable Setters

Criticality	Minor / Informative
Location	contracts/Wagon.sol#L59
Status	Unresolved

Description

The contract performs operations on variables that have been configured on user-supplied input. These variables are missing of proper check for the case where a value is zero. This can lead to problems when the contract is executed, as certain actions may not be properly handled when the value is zero.

```
emergencyAccount = _newEmergencyAddress;
```

Recommendation

By adding the proper check, the contract will not allow the variables to be configured with zero value. This will ensure that the contract can handle all possible input values and avoid unexpected behavior or errors. Hence, it can help to prevent the contract from being exploited or operating unexpectedly.

L19 - Stable Compiler Version

Criticality	Minor / Informative
Location	contracts/Wagon.sol#L2
Status	Unresolved

Description

The `^` symbol indicates that any version of Solidity that is compatible with the specified version (i.e., any version that is a higher minor or patch version) can be used to compile the contract. The version lock is a mechanism that allows the author to specify a minimum version of the Solidity compiler that must be used to compile the contract code. This is useful because it ensures that the contract will be compiled using a version of the compiler that is known to be compatible with the code.

```
pragma solidity ^0.8.9;
```

Recommendation

The team is advised to lock the pragma to ensure the stability of the codebase. The locked pragma version ensures that the contract will not be deployed with an unexpected version. An unexpected version may produce vulnerabilities and undiscovered bugs. The compiler should be configured to the lowest version that provides all the required functionality for the codebase. As a result, the project will be compiled in a well-tested LTS (Long Term Support) environment.

Contract Functions

Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
AccessControl	Implementation	Context, IAccessControl, ERC165		
	supportsInterface	Public		-
	hasRole	Public		-
	_checkRole	Internal		
	_checkRole	Internal		
	getRoleAdmin	Public		-
	grantRole	Public	✓	onlyRole
	revokeRole	Public	✓	onlyRole
	renounceRole	Public	✓	-
	_setupRole	Internal	✓	
	_setRoleAdmin	Internal	✓	
	_grantRole	Internal	✓	
	_revokeRole	Internal	✓	
IAccessControl	Interface			
	hasRole	External		-
	getRoleAdmin	External		-
	grantRole	External	✓	-
	revokeRole	External	✓	-
	renounceRole	External	✓	-
IVotes	Interface			

	getVotes	External		-
	getPastVotes	External		-
	getPastTotalSupply	External		-
	delegates	External		-
	delegate	External	✓	-
	delegateBySig	External	✓	-
Pausable	Implementation	Context		
		Public	✓	-
	paused	Public		-
	_requireNotPaused	Internal		
	_requirePaused	Internal		
	_pause	Internal	✓	whenNotPaused
	_unpause	Internal	✓	whenPaused
ERC20	Implementation	Context, IERC20, IERC20Metadata		
		Public	✓	-
	name	Public		-
	symbol	Public		-
	decimals	Public		-
	totalSupply	Public		-
	balanceOf	Public		-
	transfer	Public	✓	-
	allowance	Public		-
	approve	Public	✓	-
	transferFrom	Public	✓	-
	increaseAllowance	Public	✓	-
	decreaseAllowance	Public	✓	-

	_transfer	Internal	✓	
	_mint	Internal	✓	
	_burn	Internal	✓	
	_approve	Internal	✓	
	_spendAllowance	Internal	✓	
	_beforeTokenTransfer	Internal	✓	
	_afterTokenTransfer	Internal	✓	
ERC20Permit	Implementation	ERC20, IERC20Permit, EIP712		
		Public	✓	EIP712
	permit	Public	✓	-
	nonces	Public		-
	DOMAIN_SEPARATOR	External		-
	_useNonce	Internal	✓	
IERC20Permit	Interface			
	permit	External	✓	-
	nonces	External		-
	DOMAIN_SEPARATOR	External		-
ERC20Burnable	Implementation	Context, ERC20		
	burn	Public	✓	-
	burnFrom	Public	✓	-
ERC20Snapshot	Implementation	ERC20		
	_snapshot	Internal	✓	
	_getCurrentSnapshotId	Internal		
	balanceOfAt	Public		-

	totalSupplyAt	Public		-
	_beforeTokenTransfer	Internal	✓	
	_valueAt	Private		
	_updateAccountSnapshot	Private	✓	
	_updateTotalSupplySnapshot	Private	✓	
	_updateSnapshot	Private	✓	
	_lastSnapshotId	Private		
ERC20Votes	Implementation	IVotes, ERC20Permit		
	checkpoints	Public		-
	numCheckpoints	Public		-
	delegates	Public		-
	getVotes	Public		-
	getPastVotes	Public		-
	getPastTotalSupply	Public		-
	_checkpointsLookup	Private		
	delegate	Public	✓	-
	delegateBySig	Public	✓	-
	_maxSupply	Internal		
	_mint	Internal	✓	
	_burn	Internal	✓	
	_afterTokenTransfer	Internal	✓	
	_delegate	Internal	✓	
	_moveVotingPower	Private	✓	
	_writeCheckpoint	Private	✓	
	_add	Private		
	_subtract	Private		

IERC20Metadata	Interface	IERC20		
	name	External		-
	symbol	External		-
	decimals	External		-
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
Arrays	Library			
	findUpperBound	Internal		
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		
Counters	Library			
	current	Internal		
	increment	Internal	✓	
	decrement	Internal	✓	
	reset	Internal	✓	
EIP712	Implementation			
		Public	✓	-
	_domainSeparatorV4	Internal		

	_buildDomainSeparator	Private		
	_hashTypedDataV4	Internal		
ECDSA	Library			
	_throwError	Private		
	tryRecover	Internal		
	recover	Internal		
	tryRecover	Internal		
	recover	Internal		
	tryRecover	Internal		
	recover	Internal		
	toEthSignedMessageHash	Internal		
	toEthSignedMessageHash	Internal		
	toTypedDataHash	Internal		
ERC165	Implementation	IERC165		
	supportsInterface	Public		-
IERC165	Interface			
	supportsInterface	External		-
Math	Library			
	max	Internal		
	min	Internal		
	average	Internal		
	ceilDiv	Internal		
	mulDiv	Internal		
	mulDiv	Internal		
	sqrt	Internal		

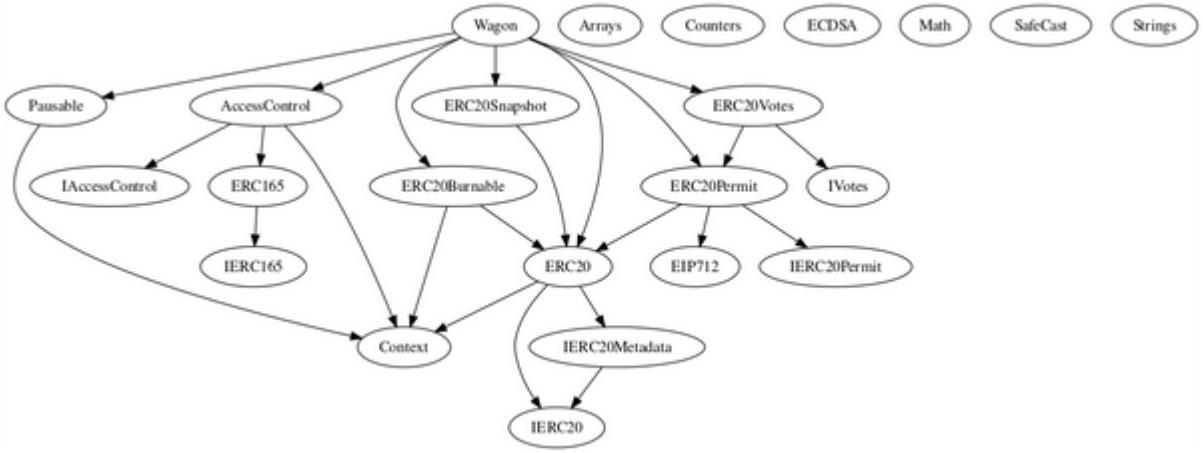
	sqrt	Internal		
SafeCast	Library			
	toUint248	Internal		
	toUint240	Internal		
	toUint232	Internal		
	toUint224	Internal		
	toUint216	Internal		
	toUint208	Internal		
	toUint200	Internal		
	toUint192	Internal		
	toUint184	Internal		
	toUint176	Internal		
	toUint168	Internal		
	toUint160	Internal		
	toUint152	Internal		
	toUint144	Internal		
	toUint136	Internal		
	toUint128	Internal		
	toUint120	Internal		
	toUint112	Internal		
	toUint104	Internal		
	toUint96	Internal		
	toUint88	Internal		
	toUint80	Internal		
	toUint72	Internal		
	toUint64	Internal		
	toUint56	Internal		
	toUint48	Internal		

	toUint40	Internal		
	toUint32	Internal		
	toUint24	Internal		
	toUint16	Internal		
	toUint8	Internal		
	toUint256	Internal		
	toInt248	Internal		
	toInt240	Internal		
	toInt232	Internal		
	toInt224	Internal		
	toInt216	Internal		
	toInt208	Internal		
	toInt200	Internal		
	toInt192	Internal		
	toInt184	Internal		
	toInt176	Internal		
	toInt168	Internal		
	toInt160	Internal		
	toInt152	Internal		
	toInt144	Internal		
	toInt136	Internal		
	toInt128	Internal		
	toInt120	Internal		
	toInt112	Internal		
	toInt104	Internal		
	toInt96	Internal		
	toInt88	Internal		
	toInt80	Internal		
	toInt72	Internal		

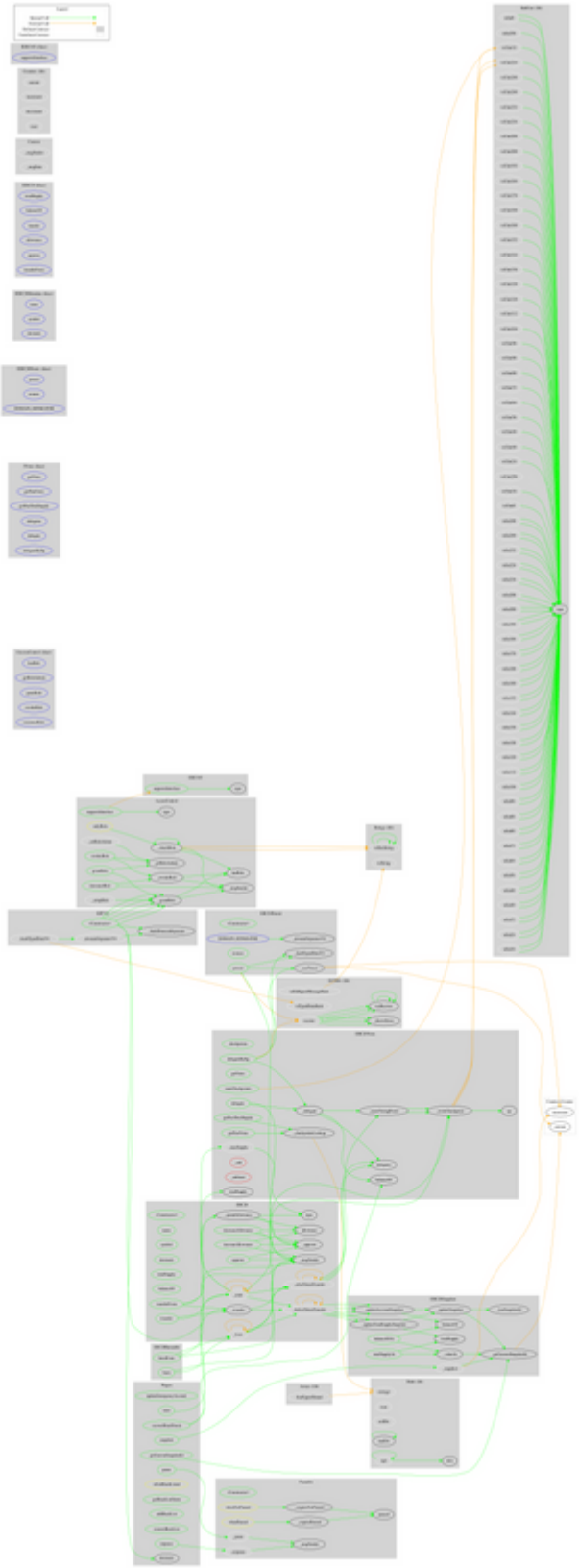
	toInt64	Internal		
	toInt56	Internal		
	toInt48	Internal		
	toInt40	Internal		
	toInt32	Internal		
	toInt24	Internal		
	toInt16	Internal		
	toInt8	Internal		
	toInt256	Internal		
Strings	Library			
	toString	Internal		
	toHexString	Internal		
	toHexString	Internal		
	toHexString	Internal		
Wagon	Implementation	ERC20, ERC20Burnable, ERC20Snapshot, AccessControl, Pausable, ERC20Permit, ERC20Votes		
		Public	✓	ERC20 ERC20Permit
	updateEmergencyAccount	Public	✓	onlyRole
	snapshot	Public	✓	onlyRole
	getCurrentSnapshotId	Public		-
	pause	Public	✓	onlyRole
	unpause	Public	✓	onlyRole

	mint	Public	✓	onlyRole
	_beforeTokenTransfer	Internal	✓	whenNotPaused isNotBlackListed
	getBlackListStatus	Public		-
	addBlackList	Public	✓	onlyRole
	removeBlackList	Public	✓	onlyRole
	recoverBlackFunds	Public	✓	onlyRole
	_afterTokenTransfer	Internal	✓	
	_mint	Internal	✓	
	_burn	Internal	✓	

Inheritance Graph



Flow Graph



Summary

There are some functions that can be abused by the owner like stop transactions, transfer the user's tokens, mint tokens and blacklist addresses. if the contract owner abuses the mint functionality, then the contract will be highly inflated. A multi-wallet signing pattern will provide security against potential hacks. Temporarily locking the contract or renouncing ownership will eliminate all the contract threats.

Disclaimer

The information provided in this report does not constitute investment, financial or trading advice and you should not treat any of the document's content as such. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes nor may copies be delivered to any other person other than the Company without Cyberscope’s prior written consent. This report is not nor should be considered an “endorsement” or “disapproval” of any particular project or team. This report is not nor should be regarded as an indication of the economics or value of any “product” or “asset” created by any team or project that contracts Cyberscope to perform a security assessment. This document does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors' business, business model or legal compliance. This report should not be used in any way to make decisions around investment or involvement with any particular project. This report represents an extensive assessment 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 Cyberscope’s position is that each company and individual are responsible for their own due diligence and continuous security Cyberscope’s goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies and in no way claims any guarantee of security or functionality of the technology we agree to analyze. The assessment services provided by Cyberscope are subject to dependencies and are under continuing development. You agree that your access and/or use including but not limited to any services reports and materials will be at your sole risk on an as-is where-is and as-available basis Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives false negatives and other unpredictable results. The services may access and depend upon multiple layers of third parties.

About Cyberscope

Cyberscope is a blockchain cybersecurity company that was founded with the vision to make web3.0 a safer place for investors and developers. Since its launch, it has worked with thousands of projects and is estimated to have secured tens of millions of investors' funds.

Cyberscope is one of the leading smart contract audit firms in the crypto space and has built a high-profile network of clients and partners.



The Cyberscope team

https://www.cyberscope.io