Cyberscope

Audit Report

Tokenee

October 2023

SHA256 792befeb8ff234f94586be73cd235e76f9b73c799201001617ebbdc9b301284e

Audited by © cyberscope

Table of Contents

Review

Testing Deploy	https://testnet.bscscan.com/address/0xfc6970544e8200924507b825448244f5aced1f18

Audit Updates

Initial Audit	09 Oct 2023

Source Files

Filename	SHA256
contracts/AirdropDistributor.sol	792befeb8ff234f94586be73cd235e76f9b73c799201001617ebbdc9b301284e
contracts/interfaces/IAirdropDistributor.sol	1d92c64de669038ee4a4f6526eb7859a565ded06002d14f5ca550e4d0f512963
@openzeppelin/contracts/utils/Strings.sol	cb2df477077a5963ab50a52768cb74ec6f32177177a78611ddbbe2c07e2d36de
@openzeppelin/contracts/utils/Context.sol	1458c260d010a08e4c20a4a517882259a23a4baa0b5bd9add9fb6d6a1549814a
@openzeppelin/contracts/utils/Address.sol	8b85a2463eda119c2f42c34fa3d942b61aee65df381f48ed436fe8edb3a7d602
@openzeppelin/contracts/utils/math/SignedMath.sol	420a5a5d8d94611a04b39d6cf5f02492552ed4257ea82aba3c765b1ad52f77f6
@openzeppelin/contracts/utils/math/Math.sol	85a2caf3bd06579fb55236398c1321e15fd524a8fe140dff748c0f73d7a52345
@openzeppelin/contracts/utils/introspection/IERC165.sol	701e025d13ec6be09ae892eb029cd83b3064325801d73654847a5fb11c58b1e5

@openzeppelin/contracts/utils/introspection/ERC165.sol	8806a632d7b656cadb8133ff8f2acae4405b3a64d8709d93b0fa6a216a8a6154
@openzeppelin/contracts/token/ERC20/IERC20.sol	7ebde70853ccafcf1876900dad458f46eb9444d591d39bfc58e952e2582f5587
@openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol	82dc918d8df553e2461b96595580e565424b407d73dab8a7ce4bde479810fb2a
@openzeppelin/contracts/token/ERC20/extensions/IERC20Permit.sol	b7383c48331f3cc9901fc05e5d5830fcd533699a77f3ee1e756a98681bfbb2ee
@openzeppelin/contracts/access/IAccessControl.sol	d03c1257f2094da6c86efa7aa09c1c07ebd33dd31046480c5097bc2542140e45
@openzeppelin/contracts/access/AccessControl.sol	afd98330d27bddff0db7cb8fcf42bd4766dda5f60b40871a3bec6220f9c9edf7

Overview

The AirdropDistributor contract is designed to facilitate the distribution of rewards in the form of a specific ERC-20 token, which is specified by the `rewardToken` variable. The key functionality of this contract includes the ability to transfer rewards to a group of investors defined in the `rewardData` array, with checks for valid parameters such as airdrop ID, treasury address, allocated budget, and investor details. The contract ensures that the total rewards transferred do not exceed the specified budget. Lastly, the contract enforces access control, ensuring that only operators with the `OPERATOR_ROLE` can execute certain functions, making it suitable for managing airdrop campaigns securely.

Roles

Operator

The operator has the authority to:

- transfer rewards to investors by calling the `transferRewards` function.
- update the reward token address using the `setRewardToken` function.

Findings Breakdown

Critical	0
Medium	0
Minor / Informative	3

Severity	Unresolved	Acknowledged	Resolved	Other
Critical	0	0	0	0
Medium	0	0	0	0
Minor / Informative	3	0	0	0

Diagnostics

Critical Medium Minor / Informative

Severity	Code	Description	Status
●	CI	Contract-Interface Inconsistency	Unresolved
●	PTAI	Potential Transfer Amount Inconsistency	Unresolved
●	FVO	Function Visibility Optimization	Unresolved

CI - Contract-Interface Inconsistency

Criticality	Minor / Informative
Location	contracts/interfaces/IAirdropDistributor.sol#L34 contracts/AirdropDistributor.sol#L37
Status	Unresolved

Description

There is an inconsistency between the function visibility declared in the contract's interface and its actual implementation. The interface declares the `setRewardToken` function with external visibility, while in the contract's implementation, it is declared with public visibility. This inconsistency can lead to confusion and should be addressed for clarity and adherence to the defined interface.

```
function setRewardToken(address newToken) external;

function setRewardToken(address newRewardToken) public
onlyRole(OPERATOR_ROLE) {
    _setRewardToken(newRewardToken);
}
```

Recommendation

To ensure consistency and alignment between the contract and its interface, the team is advised to update the implementation of the `setRewardToken` function to match the visibility declared in the interface.

PTAI - Potential Transfer Amount Inconsistency

Criticality	Minor / Informative
Location	contracts/AirdropDistributor.sol#L84
Status	Unresolved

Description

The `transfer()` and `transferFrom()` functions are used to transfer a specified amount of tokens to an address. The fee or tax is an amount that is charged to the sender of an ERC20 token when tokens are transferred to another address. According to the specification, the transferred amount could potentially be less than the expected amount. This may produce inconsistency between the expected and the actual behavior.

The following example depicts the diversion between the expected and actual amount.

Tax	Amount	Expected	Actual
No Tax	100	100	100
10% Tax	100	100	90

The contract currently tracks the sum of transferred tokens in the `totalTransfered` variable within the `transferRewards` function. However, if the `rewardToken` contract has logic that includes fees or deductions during the transfer process, the `totalTransfered` variable may not accurately represent the actual total transferred amount.

```
for (uint256 i; i < length; ) {
    InvestorReward memory investor = transferData[i];
    _checkInvestorData(totalTransfered, budget, investor);
    _transferTokens(treasury, investor);
    unchecked {
        totalTransfered += investor.reward;
        ++i;
    }
}
```

Recommendation

The team is advised to take into consideration the actual amount that has been transferred instead of the expected.

It is important to note that an ERC20 transfer tax is not a standard feature of the ERC20 specification, and it is not universally implemented by all ERC20 contracts. Therefore, the contract could produce the actual amount by calculating the difference between the transfer call.

```
 Actual Transferred Amount = Balance After Transfer - Balance Before Transfer
```

FVO - Function Visibility Optimization

Criticality	Minor / Informative
Location	contracts/AirdropDistributor.sol#L37
Status	Unresolved

Description

The contract contains a `setRewardToken` function with a public visibility setting, which means it can be accessed both from within and outside the contract. However, this function is not utilized within the contract's code, making its public accessibility unnecessary.

```
function setRewardToken(address newRewardToken) public
onlyRole(OPERATOR_ROLE) {
    _setRewardToken(newRewardToken);
}
```

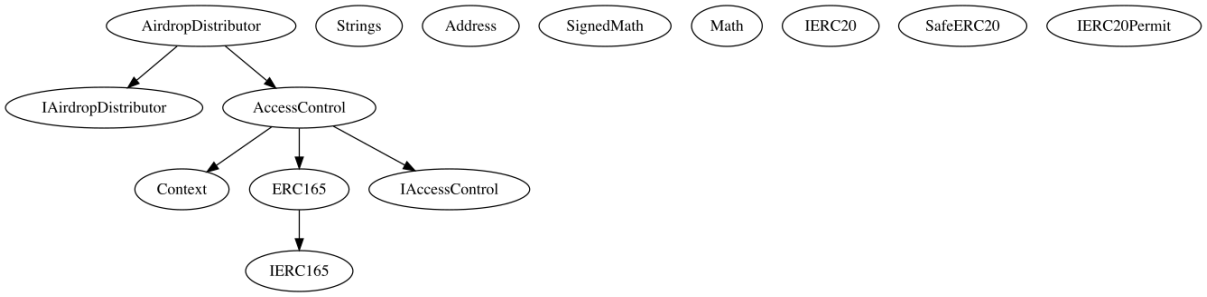
Recommendation

To enhance code clarity and security, the team is recommended to update the visibility of the `setRewardToken` function to "external" instead of "public." This change will ensure it can only be called by external contracts or addresses that need to interact with it directly. This modification aligns the function's visibility with its current usage pattern.

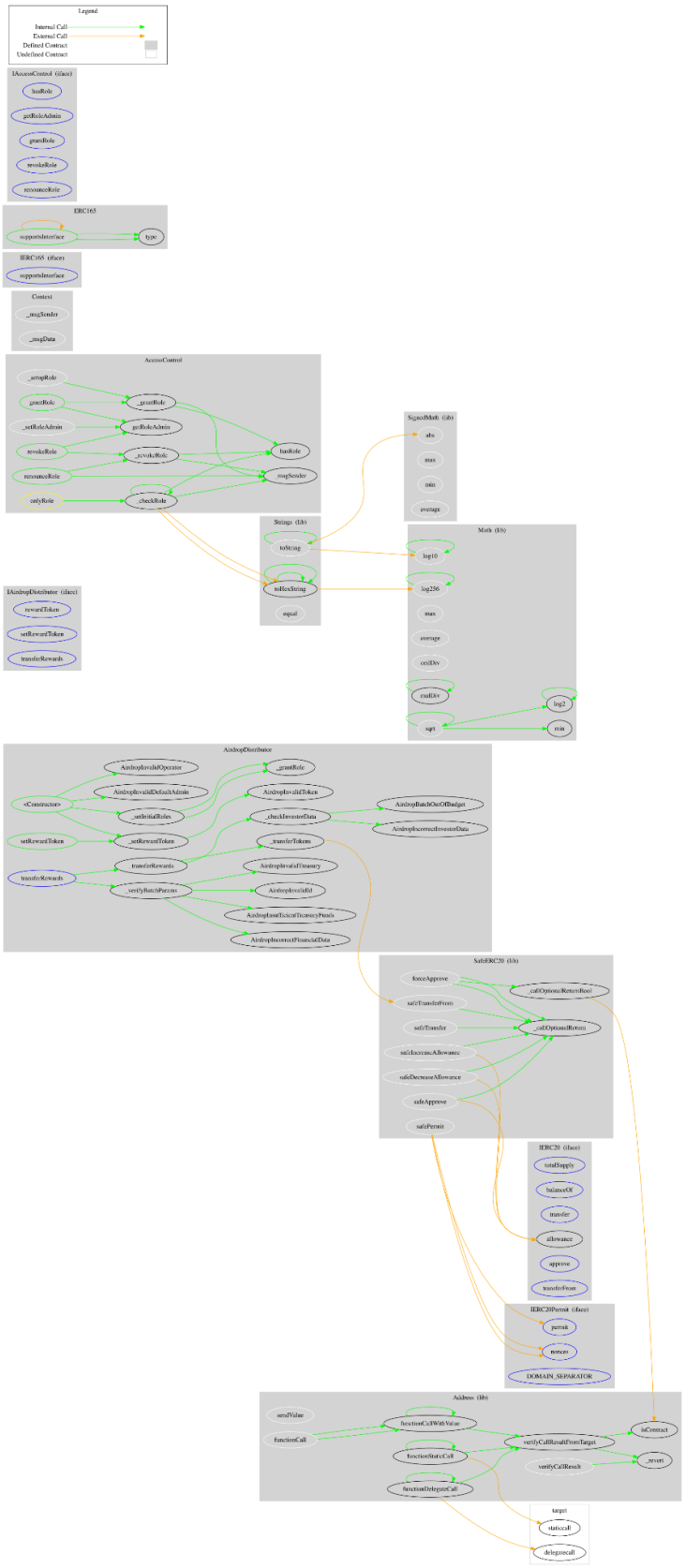
Functions Analysis

Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
AirdropDistributor	Implementation	IAirdropDistributor, AccessControl		
		Public	✓	-
	transferRewards	External	✓	onlyRole
	setRewardToken	Public	✓	onlyRole
	_setRewardToken	Internal	✓	
	_setInitialRoles	Internal	✓	
	_verifyBatchParams	Internal		
	_transferRewards	Internal	✓	
	_checkInvestorData	Internal		
	_transferTokens	Internal	✓	
IAirdropDistributor	Interface			
	rewardToken	External		-
	setRewardToken	External	✓	-
	transferRewards	External	✓	-

Inheritance Graph



Flow Graph


Legend
Internal Call
External Call
Defined Contract
Undefined Contract
IAccessControl (iface)
hasRole
getRoleAdmin
grantRole
revokeRole
renounceRole
ERC165
supportsInterface
type
IERC165 (iface)
supportsInterface
Context
_msgSender
_msgData
AccessControl
_setupRole
grantRole
_setRoleAdmin
revokeRole
renounceRole
onlyRole
_grantRole
getRoleAdmin
_revokeRole
_checkRole
hasRole
_msgSender
SignedMath (Lib)
abs
max
min
average
Strings (Lib)
toString
toHexString
equal
Math (Lib)
log10
log256
max
average
ceilDiv
mulDiv
log2
sqrt
min
IAirdropDistributor (iface)
rewardToken
setRewardToken
transferRewards
AirdropDistributor
AirdropInvalidOperator
AirdropInvalidDefaultAdmin
«Constructor»
_setInitialRoles
setRewardToken
_setRewardToken
transferRewards
transferRewards
_verifyBatchParams
_grantRole
AirdropInvalidToken
_checkInvestorData
AirdropBatchOutOfBudget
AirdropIncorrectInvestorData
_transferTokens
AirdropInvalidTreasury
AirdropInvalidId
AirdropTotalTokenTreasuryPoints
AirdropIncorrectFinancialData
SafeERC20 (Lib)
forceApprove
safeTransferFrom
safeTransfer
safeIncreaseAllowance
safeDecreaseAllowance
safeApprove
safePermit
_callOptionalReturnBool
_callOptionalReturn
IERC20 (iface)
totalSupply
balanceOf
transfer
allowance
approve
transferFrom
IERC20Permit (iface)
permit
nonces
DOMAIN_SEPARATOR
Address (Lib)
sendValue
functionCall
functionCallWithValue
functionStaticCall
functionDelegateCall
verifyCallResultFromTarget
verifyCallResult
isContract
_revert
target
staticcall
delegatecall

Summary

Tokenee contract implements a rewards and utility mechanism. This audit investigates security issues, business logic concerns and potential improvements.

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