Cyberscope

Audit Report

VestorFactory

October 2022

Github	https://github.com/vestor-co/vestor-contracts
Commit	9814dd933047ace2f99bf56e9a239bfe09ff785e
Audited by	© cyberscope

Table of Contents

Contract Review

Contract Name	vestor
Compiler Version	v0.8.11+commit.d7f03943
Github	https://github.com/vestor-co/vestor-contracts
Commit	9814dd933047ace2f99bf56e9a239bfe09ff785e

Audit Updates

Initial Audit	12th October 2022
Corrected	

Source Files

Filename	SHA256
@openzeppelin/contracts-upgradeable/proxy/utils/Initializable.sol	cd823c76cbf5f5b6ef1bda565d58be66c843c37707cd93eb8fb5425deebd6756
@openzeppelin/contracts-upgradeable/utils/AddressUpgradeable.sol	35fb271561f3dc72e91b3a42c6e40c2bb2e788cd8ca58014ac43f6198b8d32ca
@openzeppelin/contracts-upgradeable/utils/CountersUpgradeable.sol	5c1ac829a429b0c2ca9b4c9ed8b78d412320e9175e45f088c4e9056ef95fbf21
@openzeppelin/contracts/access/Ownable.sol	9353af89436556f7ba8abb3f37a6677249aa4df6024fbfaa94f79ab2f44f3231
@openzeppelin/contracts/proxy/Clones.sol	c996327afa2e09915e830724f7c88c900d4a5ba2d456dd1c55f955614d1fd3e9
@openzeppelin/contracts/security/ReentrancyGuard.sol	aa73590d5265031c5bb64b5c0e7f84c44cf5f8539e6d8606b763adac784e8b2e
@openzeppelin/contracts/token/ERC1155/ERC1155.sol	3a7b1481259da24728a0bac33ac9728c0faf71d436e4f198209815f732240a24
@openzeppelin/contracts/token/ERC1155/extensions/ERC1155Burnable.sol	1093c31ab9989866598a66e0d162d63aeae7e008c9cdf2b6625f113d6e30ae2b
@openzeppelin/contracts/token/ERC1155/extensions/IERC1155MetadataURI.sol	6987fbfa647d3da51e8c270371ac48c5fcd26fb046cf54644b39aa098ae30324

@openzeppelin/contracts/token/ERC1155/IERC1155.sol	fd6a1801f1f2f8af0a3ece0b254da06ec24568aec02cfe94827061379aebc6f3
@openzeppelin/contracts/token/ERC1155/IERC1155Receiver.sol	578834a1bcdac6a22de5e07ae63bbbd4d41615f35950afc6e6c068d92619b334
@openzeppelin/contracts/token/ERC20/IERC20.sol	94f23e4af51a18c2269b355b8c7cf4db8003d075c9c541019eb8dcf4122864d5
@openzeppelin/contracts/utils/Address.sol	1e0922f6c0bf6b1b8b4d480dcabb691b1359195a297bde6dc5172e79f3a1f826
@openzeppelin/contracts/utils/Context.sol	1458c260d010a08e4c20a4a517882259a23a4baa0b5bd9add9fb6d6a1549814a
@openzeppelin/contracts/utils/introspection/ERC165.sol	8806a632d7b656cadb8133ff8f2acae4405b3a64d8709d93b0fa6a216a8a6154
@openzeppelin/contracts/utils/introspection/IERC165.sol	701e025d13ec6be09ae892eb029cd83b3064325801d73654847a5fb11c58b1e5
contracts/vestor.sol	8e96043137567458e08a12ac5e0f4609d31cf84823f1c639fc120a8e5127d805
contracts/vestorFactory.sol	0ac3f06931fcdab4d4e142b816e3ddc99a5f6c074ede49b205c361c22aceb97b

Introduction

The VestorFactory contract is responsible for creating and configuring the vesting contracts. Additionally, the factory contract monitors the vesting contracts by keeping two registries.

1. A list of the vesting contracts.
2. A list of vesting contracts in which each investor participates.

The factory contract taxes each vesting contract creation. The fee can either be fixed in native currency or proportionally to the token's vesting amount.

Roles

The contract owner has the ability to manipulate the above mentioned fees.

Contract Diagnostics

● Critical ● Medium ● Minor / Informative

Severity	Code	Description	Status
●	ELFM	Exceeds Fees Limit Manipulation	Unresolved
●	STC	Succeeded Transfer Check	Unresolved
●	MC	Missing Check	Unresolved
●	MDM	More Descriptive Messages	Unresolved
●	FPI	Fee Precision Issue	Unresolved
●	L01	Public Function could be Declared External	Unresolved
●	L04	Conformance to Solidity Naming Conventions	Unresolved
●	L07	Missing Events Arithmetic	Unresolved

ELFM - Exceeds Fees Limit Manipulation

Criticality	minor/informative
Location	contract.sol#L77
Status	Unresolved

Description

The contract owner has the authority to increase the fees by over 100%. The owner may take advantage of it by calling the updatetokenfeespercentage functions with a value higher than 10000. This is not a major issue but since the contract divides the `tokenees` by the number 10000, then intuitively the `tokenfees` should not be more than 10000.

```
function updatetokenfeespercentage(uint256 _percentageinbp)public onlyOwner{
    tokenfees = _percentageinbp;
}
```

Recommendation

The contract could embody a check for the maximum acceptable value. The maximum acceptable value shouldn't exceed 10000.

MC - Missing Check

Criticality	minor / informative
Location	contract.sol#L21,37,40
Status	Unresolved

Description

The contract is processing variables that have not been properly sanitized and checked that they form the proper shape. These variables may produce vulnerability issues. To be more specific the variable `startperiod` and `cliffperiod` are not properly sanitized.

```
function _clone(
    address impl,
    string memory name,
    address tokencontractaddress,
    address[]calldata investors,
    uint256 vestingPeriod,
    uint256[]calldata amountperinvestors,
    uint256 startperiod,
    uint256 cliffperiod
```

The contract is making transactions that have not been properly sanitized. These transactions may produce issues and revert. Also, the contract does not check if enough tokens are invested to start vesting.

```
uint256 totalamount =
vestor(clone).addforinvestors(amountperinvestors,vestingPeriod,cliffperiod);
require(msg.value >= contractfees ||
IERC20(tokencontractaddress).transferFrom(msg.sender,address(this),totalamount*tokenfees/100
00));


IERC20(tokencontractaddress).transferFrom(msg.sender,clone,totalamount);
```

Recommendation

The contract should properly check the variables according to the required specifications.

- Startperiod should be greater or equal to the current block time.
- Cliffperiod should be greater than zero.
- Totalamount should be greater than zero.

It is recommended to pre-check if the investor has the available balance. To fulfill those transactions.

STC - Succeeded Transfer Check

Criticality	minor / informative
Location	contract.sol#40
Status	Unresolved

Description

According to the ERC20 specification, the transfer methods should be checked if the result is successful. Otherwise, the contract may wrongly assume that the transfer has been established.

```
IERC20(tokencontractaddress).transferFrom(msg.sender,clone,totalamount);
```

Recommendation

The contract should check if the result of the transfer methods is successful.

MDM - More Descriptive Messages

Criticality	minor / informative
Location	contract.sol#L21,37,40
Status	Unresolved

Description

The contract does not provide an explanation about what went wrong with the require statements.

```
require(msg.value >= contractfees ||
IERC20(tokencontractaddress).transferFrom(msg.sender,address(this),totalamount
*tokenfees/10000));
```

Recommendation

It is recommended to provide an explanation about what went wrong with the require statements. Also, it is recommended to divide the checks into different require statements. For instance:

- require(msg.value >= contractfees, “Not enough ETH”)
- (if msg.value is zero) require(IERC20(tokencontractaddress).transferFrom.., “Not enough tokens”)

FPI - Fee Precision Issue

Criticality	minor / informative
Location	contract.sol#L37,87
Status	Unresolved

Description

In solidity all integer division rounds down to the nearest integer. That means that if the totalamount*tokenfees is lower than 10000. No fee will be accumulated in the contract. For instance,

Variables	Value
totalamount	900
tokenfees	10
Fee = totalamount*tokenfees/10000	900*10/10000 -> 0.9 -> 0

```
require(msg.value >= contractfees ||
IERC20(tokencontractaddress).transferFrom(msg.sender,address(this),totalamount*tokenfees/1
0000));

function getTokenfee(uint256 _amount)public view returns (uint256){
        return _amount*tokenfees/10000;
    }
```

Recommendation

The zero rounding is an expected result when the values are less than a threshold. We state that the percentage taxes are necessary for calculating the fees. Thus, we emphasise the team should be aware of potential losses.

L01 - Public Function could be Declared External

Criticality	minor / informative
Location	contracts/vestorFactory.sol#L69,73,61,21,77,65,81,85
Status	Unresolved

Description

Public functions that are never called by the contract should be declared external to save gas.

```
fetchcontractscreated
updatetokenfeespercentage
fetchaddress
_clone
updatecontractfeespercentage
fetchinvaddress
getTokenfee
getTokencontractfee
```

Recommendation

Use the external attribute for functions never called from the contract.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor / informative
Location	contracts/vestorFactory.sol#L73,65,9,11,77,21,61,81
Status	Unresolved

Description

Solidity defines a naming convention that should be followed. Rule exceptions:

- Allow constant variable name/symbol/decimals to be lowercase.
- Allow _ at the beginning of the mixed_case match for private variables and unused parameters.

```
_percentageinbp
_address
vestorFactory
proxydeployed
_fees
_clone
_amount
```

Recommendation

Follow the Solidity naming convention.
https://docs.soliditylang.org/en/v0.4.25/style-guide.html#naming-conventions.

L07 - Missing Events Arithmetic

Criticality	minor / informative
Location	contracts/vestorFactory.sol#L73,77
Status	Unresolved

Description

Detected missing events for critical arithmetic parameters. There are functions that have no event emitted, so it is difficult to track off-chain changes.

```
tokenfees = _percentageinbp
contractfees = _fees
```

Recommendation

Emit an event for critical parameter changes.

Contract Functions

Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
Initializable	Implementation			
	_disableInitializers	Internal	✓	
AddressUpgradeable	Library			
	isContract	Internal		
	sendValue	Internal	✓	
	functionCall	Internal	✓	
	functionCall	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionStaticCall	Internal		
	functionStaticCall	Internal		
	verifyCallResult	Internal		
CountersUpgradeable	Library			
	current	Internal		
	increment	Internal	✓	
	decrement	Internal	✓	
	reset	Internal	✓	
Ownable	Implementation	Context		
	<Constructor>	Public	✓	-
	owner	Public		-
	_checkOwner	Internal		
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
	_transferOwnership	Internal	✓	

Clones	Library			
	clone	Internal	✓	
	cloneDeterministic	Internal	✓	
	predictDeterministicAddress	Internal		
	predictDeterministicAddress	Internal		
ReentrancyGuard	Implementation			
	<Constructor>	Public	✓	-
ERC1155	Implementation	Context, ERC165, IERC1155, IERC1155MetadataURI		
	<Constructor>	Public	✓	-
	supportsInterface	Public		-
	uri	Public		-
	balanceOf	Public		-
	balanceOfBatch	Public		-
	setApprovalForAll	Public	✓	-
	isApprovedForAll	Public		-
	safeTransferFrom	Public	✓	-
	safeBatchTransferFrom	Public	✓	-
	_safeTransferFrom	Internal	✓	
	_safeBatchTransferFrom	Internal	✓	
	_setURI	Internal	✓	
	_mint	Internal	✓	
	_mintBatch	Internal	✓	
	_burn	Internal	✓	
	_burnBatch	Internal	✓	
	_setApprovalForAll	Internal	✓	
	_beforeTokenTransfer	Internal	✓	
	_afterTokenTransfer	Internal	✓	
	_doSafeTransferAcceptanceCheck	Private	✓	

	_doSafeBatchTransferAcceptanceCheck	Private	✓	
	_asSingletonArray	Private		
ERC1155Burnable	Implementation	ERC1155		
	burn	Public	✓	-
	burnBatch	Public	✓	-
IERC1155MetadataURI	Interface	IERC1155		
	uri	External		-
IERC1155	Interface	IERC165		
	balanceOf	External		-
	balanceOfBatch	External		-
	setApprovalForAll	External	✓	-
	isApprovedForAll	External		-
	safeTransferFrom	External	✓	-
	safeBatchTransferFrom	External	✓	-
IERC1155Receiver	Interface	IERC165		
	onERC1155Received	External	✓	-
	onERC1155BatchReceived	External	✓	-
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
Address	Library			
	isContract	Internal		

	sendValue	Internal	✓	
	functionCall	Internal	✓	
	functionCall	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionStaticCall	Internal		
	functionStaticCall	Internal		
	functionDelegateCall	Internal	✓	
	functionDelegateCall	Internal	✓	
	verifyCallResult	Internal		
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		
ERC165	Implementation	IERC165		
	supportsInterface	Public		-
IERC165	Interface			
	supportsInterface	External		-
vestor	Implementation	Initializable, Reentrancy Guard		
	initialize	Public	✓	initializer
	vestTokens	Internal	✓	
	claimtokens	Public	✓	nonReentrant
	getContract	Public		-
	isWhitelisted	Public		-
	gettime	Internal		
	gettotalamountunlocked	Public		-
	haveContract	Internal		
	fetchcontractswhitelisted	Public		-
	getamount	Public		-
	addforamount	Public		-
	addforinvestors	Public		-

vestorFactory	Implementation	Ownable		
	_clone	Public	Payable	-
	fetchaddress	Public		-
	fetchinvaddress	Public		-
	fetchcontractscreated	Public		-
	updatetokenfeespercentage	Public	✓	onlyOwner
	updatecontractfeespercentage	Public	✓	onlyOwner
	getTokenfee	Public		-
	getTokencontractfee	Public		-

Contract Flow

Summary

The VestorFactory contract is responsible for generating vesting contract. This audit investigates security issues, mentions business logic concerns, and potential improvements.

Disclaimer

All the content provided in this document is for general information only and should not be used as financial advice or a reason to buy any investment.

Cyberscope team provides no guarantees against the sale of team tokens or the removal of liquidity by the project audited in this document. Always Do your own research and protect yourselves from being scammed.

The Cyberscope team has audited this project for general information and only expresses their opinion based on similar projects and checks from popular diagnostic tools. Under no circumstances did Cyberscope receive a payment to manipulate those results or change the awarding badge that we will be adding in our website.

Always Do your own research and protect yourselves from scams. This document should not be presented as a reason to buy or not buy any particular token.

The Cyberscope team disclaims any liability for the resulting losses.

About Cyberscope

Coinscope audit and K.Y.C. service has been rebranded to Cyberscope.

Coinscope is the leading early coin listing, voting and auditing authority firm. The audit process is analyzing and monitoring many aspects of the project. That way, it gives the community a good sense of security using an informative report and a generic score.

Cyberscope and Coinscope are aiming to make crypto discoverable and efficient globally. They provide all the essential tools to assist users draw their own conclusions.

The Cyberscope team

https://www.cyberscope.io