Cyberscope

Audit Report

FOMO DAO

April 2022

SHA256 a6fd9e5d6a136a6d54d5728e46847d71cbf17314e44d78e52654b47398395804

Audited by © cyberscope

Table of Contents

Source Files

Filename	SHA256
FomoEnumerableMap.sol	d21faf407ce9fa5a59d49b77105de0b8b3cd782f1de975c8beae4ec60029c0f3
FomoProxy.sol	a6fd9e5d6a136a6d54d5728e46847d71cbf17314e44d78e52654b47398395804

Audit Updates

Initial Audit	17th April 2022
Corrected	

Contract Analysis

- A user has the authority to set his leader.
- A user is automatically following his leader.
- The leader's leader is the marketing wallet
- The contract owner has the authority to change the game timestamps, first buyer, the fees, limits, to enable which team will be active, remove users, set marketing wallet, add team leaders
- If block protection is active, then the users cannot trade in the same block number.
- The participation amount should be more than 100 USDT and multiplied with 100, for instance, 100, 200, 300, etc.
- If the first or second team is enabled, then the sender should belong to the corresponding team.
- If a user belongs to a team, then there is a minimum $5000 threshold on the first purchase.
- Every time that a trade is taking place, the jackpot amount is raised until it reaches an upper time limit.

Add team Leader

The team index should be checked otherwise a leader will not be assigned to any team.

Enormous gas amount

During the transfer process a lot of structures are iterated and many transfers are taking place. The contract could approach a more performant way by splitting the dividends logic in a secondary contract. A similar approach is used in the contract with dividend trackers.

Redundant Payable Methods

Criticality	minor
Location	contract.sol#L219,515

Description

The contract contains payable methods that do not check the payable amount. As a result, they could be called with zero amount by the users. Hence, the payable functionality is redundant.

Recommendation

The contract could either check the payable amount or remove the payable keyword from the corresponding methods.

Previous Market Wallet Inconsistency

Criticality	minor
Location	contract.sol#L101

Description

The contract owner can configure the market wallet. The market wallet is attached to a user. The previous marketing wallet is not updated. Thus, many users with marketing wallet will be produced even if the contract will track only the latest.

```
function updateMarketWallet(address newMarketWallet) public onlyOwner {
       User memory userInfo = getUser(newMarketWallet);
       userInfo.account = newMarketWallet;
       userInfo.isBoughtLiquidity = true;
       userInfo.userType = 2;
       userInfo.ctime = block.timestamp;
       //default leader
       addUser(newMarketWallet, userInfo);
       _marketWallet = newMarketWallet;
}
```

Recommendation

The contract could update the previous marketing wallet user when a new one is set.

Infinite Recursions

Criticality	minor
Location	contract.sol#L112

Description

The contract is based on recursion in order to gather the required list. These recursions do not have a clear way to be stopped. Thus, there are cases that an infinite recursion may be produced. For instance, if two users follow each other indirectly, then the recursion will never stop.

```
function _inNftFollowerList(User[] memory userNftFollowerList, uint256 f,
address account) private view {
	User memory accountInfo = getUser(account);
	uint256 fsize = accountInfo.followers.length;
	for (uint256 j = 0; j < fsize; j++) {
		address faccount = accountInfo.followers[j];
		User memory followerInfo = getUser(faccount);
		if (followerInfo.userType >= 1) {
			userNftFollowerList[f] = followerInfo;
			f++;
		}
		_inNftFollowerList(userNftFollowerList, f, faccount);
	}
}

function getNftFollowerList(address account) public view returns (User[] memory)
{
	uint256 maxCount = (_userMap.values[account].nftFollowerCount + 1) * 5;
	User[] memory userNftFollowerList = new User[](maxCount);
	uint256 f = 0;
	_inNftFollowerList(userNftFollowerList, f, account);
	return userNftFollowerList;
}
```

Recommendation

The recursion should have a deterministic way to be stopped.

Infinite Loops

Criticality	minor
Location	contract.sol#L414,459,574,592

Description

There are deterministic cases where the loops will never be able to stop. For instance, if two leaders lead each other indirectly. As a result, the infinite loops will be produced.

```
while (true) {
	nftUpLeader = _userMap.values[nftUpLeader].leader;
	if (nftUpLeader == address(0)) {
		break;
	}
	if (_userMap.values[nftUpLeader].userType >= 1) {
		uint256 nftAmount = liquidity.mul(nftDividendFee).div(100);
		di.nftTo = nftUpLeader;
		di.nftAmount = nftAmount;
		break;
	}
}
```

Recommendation

The contract should guarantee in the setter methods that there will not be corner cases that will produce infinite loop.s

Contract Diagnostics

Critical Medium Minor

Severity	Code	Description
	STC	Succeeded Transfer Check
	FSA	Fixed Swap Address
	CO	Code Optimization
	MC	Missing Check
	L01	Public Function could be Declared External
	L02	State Variables could be Declared Constant
	L04	Conformance to Solidity Naming Conventions
	L07	Missing Events Arithmetic
	L09	Dead Code Elimination
	L13	Divide before Multiply Operation
	L14	Uninitialized Variables in Local Scope

Unlimited Liquidity Removal

Criticality	minor
Location	contract.sol#L661

Description

The contract owner has the authority to increase the `_globalFee` without limit, as a result. The entire liquidity amount will be removed from the swap pair..

```
uint256 globalLiquidity = contractLiquidity.mul(_globalFee).div(100);

removeLiquidity(globalLiquidity);
```

Recommendation

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

STC - Succeeded Transfer Check

Criticality	minor
Location	contract.sol#L344,553,612,615,680,714,875

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.

Recommendation

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

FSA - Fixed Swap Address

Criticality	minor
Location	contract.sol#L84

Description

The swap address is assigned once in the constructor and it can not be changed. The decentralized swaps sometimes create a new swap version or abandon the current. A contract that cannot change the swap address may not be able to catch-up the upgrade.

```
_uniswapV2Factory = addrs[0];
_uniswapV2Router = IUniswapV2Router02(addrs[1]);
...

_uniswapV2Pair = IUniswapV2Factory(_uniswapV2Factory).createPair(_tokenAddress,
_usdtAddress);
```

Recommendation

It could be better to allow the swap address mutation in case of future swap updates.

CO - Code Optimization

Criticality	minor
Location	contract.sol#L740

Description

The _lpMap property is solely used in the swapAndLiquify method. After the usage of the structure, the lpMap is cleaned. That means that there is no dependency between each execution. As a result:

- The _lpMap.tryGet will always return zero
- The gas amount will unnecessarily increased since it uses a class property for an operation that required a local property.

```
FomoEnumerableMap.AddressToUintMap private _lpMap;
...
(, uint256 lastSum) = _lpMap.tryGet(di.lastTo);
_lpMap.set(di.lastTo, lastSum.add(di.lastAmount));
...
for (uint256 cIndex = 0; cIndex < lpLength; cIndex++) {
       _lpMap.remove(receiverList[cIndex]);
}
```

Recommendation

Rewrite some code segments so the runtime will be more performant.

L01 - Public Function could be Declared External

Criticality	minor
Location	contract/@openzeppelin/contracts/access/Ownable.sol#L54,62 contract/FomoProxy.sol#L126,177,196,952

Description

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

```
isInTransferWhitelist
getUserIndexOfKey
removeUser
getNftFollowerList
transferOwnership
renounceOwnership
```

Recommendation

Use the external attribute for functions never called from the contract

L02 - State Variables could be Declared Constant

Criticality	minor
Location	contract/FomoProxy.sol#L62,51,32,20,56,60,53,75,59,57,58,54

Description

Constant state variables should be declared constant to save gas.

```
teamLeaderFee
queueMagnification
queueFee
nftDividendFee
nextGameStartWait
lastRewardFee
jackpotFee
inviterFee
_uniswapV2Pair
...
```

Recommendation

Add the constant attribute to state variables that never change.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor
Location	contract/@uniswap/v2-periphery/contracts/interfaces/IUniswapV2Router01.sol#L5 contract/FomoProxy.sol#L18,19,20,21,22,41,42,43,51,62,65,67,69,70,71

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.

```
_jackpotAddress
_nextJackpotTimestamp
_jackpotMax
_jackpotWait
_jackpotReward
HUNDRED_USDT
_nftBoughtFollowerLimit
_globalFee
_globalTriggerReward
...
```

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
Location	contract/FomoProxy.sol#L503

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.

```
gameStartTimestamp = nextTimestamp
```

Recommendation

Emit an event for critical parameter changes.

L09 - Dead Code Elimination

Criticality	minor
Location	contract/@openzeppelin/contracts/utils/structs/EnumerableSet.sol#L54,130,109,116,72,142,262,196,335,241,175,314,248,182,321,234,168,307,274,208,347 contract/FomoEnumerableMap.sol#L102,81,127,139,88,73,59,111,296,202,276,182,319,225,235,283,189,269,175,256,162,307,213

Description

Functions that are not used in the contract, and make the code's size bigger.

```
tryGet
set
remove
length
get
contains
at
_get
values
...
```

Recommendation

Remove unused functions.

L13 - Divide before Multiply Operation

Criticality	minor
Location	contract/FomoProxy.sol#L742

Description

Performing divisions before multiplications may cause lose of prediction.

```
tokenPercent = newTokenBalance.div(lpSize).div(lpAvg)
```

Recommendation

The multiplications should be prior to the divisions.

L14 - Uninitialized Variables in Local Scope

Criticality	minor
Location	contract/FomoProxy.sol#L644

Description

The are variables that are defined in the local scope and are not initialized.

```
totalSales
```

Recommendation

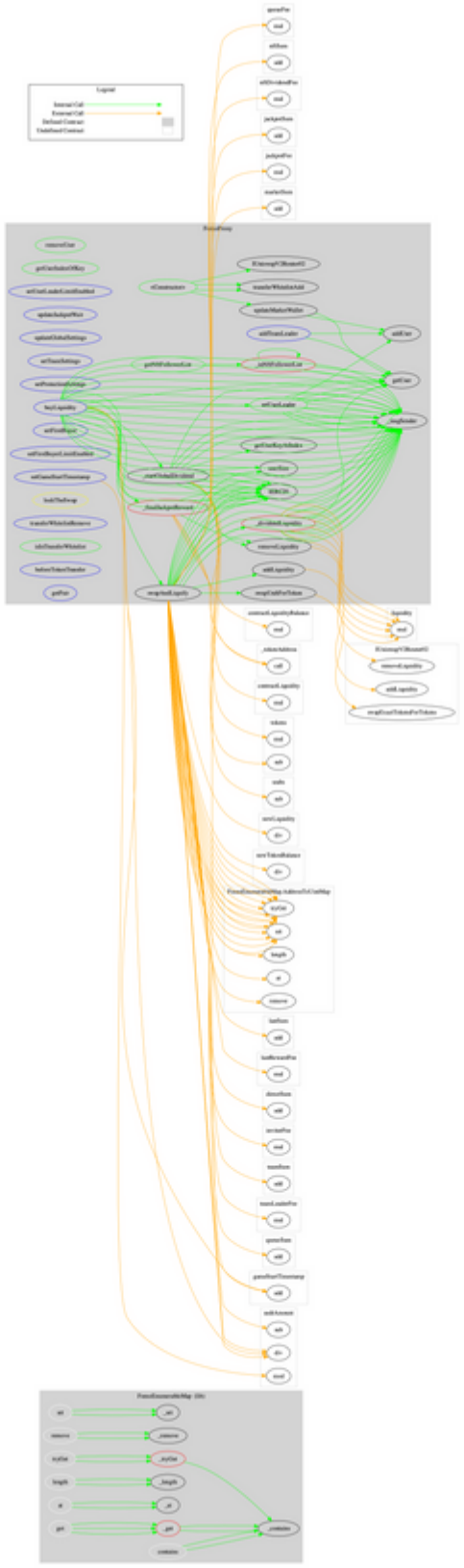
All the local scoped variables should be initialized.

Contract Functions

Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
FomoEnumerableMap	Library			
	_set	Private	✓	
	_remove	Private	✓	
	_contains	Private		
	_length	Private		
	_at	Private		
	_tryGet	Private		
	_get	Private		
	_get	Private		
	set	Internal	✓	
	remove	Internal	✓	
	contains	Internal		
	length	Internal		
	at	Internal		
	tryGet	Internal		
	get	Internal		
	get	Internal		
	set	Internal	✓	
	remove	Internal	✓	
	contains	Internal		
	length	Internal		
	at	Internal		
	tryGet	Internal		
	get	Internal		
FomoProxy	Implementation	Ownable		
	<Constructor>	Public	Payable	-
	updateMarketWallet	Public	✓	onlyOwner

	_inNftFollowerList	Private		
	getNftFollowerList	Public		-
	addUser	Private	✓	
	getUser	Public		-
	removeUser	Public	✓	onlyOwner
	getUserIndexOfKey	Public		-
	getUserKeyAtIndex	Public		-
	userSize	Public		-
	setUserLeaderLimitEnabled	External	✓	onlyOwner
	setUserLeader	Public	Payable	-
	addTeamLeader	External	✓	onlyOwner
	updateJackpotWait	External	✓	onlyOwner
	updateGlobalSettings	External	✓	onlyOwner
	setTeamSettings	External	✓	onlyOwner
	_finalJackpotReward	Private	✓	
	_dividendLiquidity	Private	✓	
	setProtectionSettings	External	✓	onlyOwner
	setGameStartTimestamp	External	✓	onlyOwner
	setFirstBuyer	External	✓	onlyOwner
	setFirstBuyerLimitEnabled	External	✓	onlyOwner
	buyLiquidity	External	Payable	-
	_startGlobalDividend	Private	✓	
	swapAndLiquify	Private	✓	lockTheSwap
	swapUsdtForToken	Private	✓	
	addLiquidity	Private	✓	
	removeLiquidity	Private	✓	
	transferWhitelistAdd	Public	✓	onlyOwner
	transferWhitelistRemove	External	✓	onlyOwner
	isInTransferWhitelist	Public		-
	beforeTokenTransfer	External		-
	getPair	External		-

Contract Flow



Domain Info

Domain Name	fomo-dao.com
Registry Domain ID	2683703996_DOMAIN_COM-VRSN
Creation Date	2022-03-23T16:27:34Z
Updated Date	2022-04-08T13:06:36Z
Registry Expiry Date	2023-03-23T16:27:34Z
Registrar WHOIS Server	whois.registrar.eu
Registrar URL	http://www.registrar.eu
Registrar	Hosting Concepts B.V. d/b/a Registrar.eu
Registrar IANA ID	1647

The domain has been created 25 days before the creation of the audit. It will expire in 11 months.

There is no public billing information, the creator is protected by the privacy settings.

Summary

FOMO DAO is a custom made contract. The implementation contains a lot of cases where could be improved. The contract should guarantee that it’s not able to produce infinite loops and endless recursions. This audit focuses on the business logic and many cases that may produce an unexpected state.

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 provides all the essential tools to assist users draw their own conclusions.

The Cyberscope team

https://www.cyberscope.io