Smart Contract
Security Audit
V1

OAF Token Smart Contract

19/8/2022

https://saferico.com/

business@saferico.com
https://t.me/SFI_ANN

Table of Contents

Background

The purpose of the audit was to achieve the following:

- Ensure that the smart contract functions as intended.
- Identify potential security issues with the smart contract.

The information in this report should be used to understand the risk exposure of the smart contract, and as a guide to improve the security posture of the smart contract by remediating the issues that were identified.

Project Information

- **Platform**: Ethereum
- **Contract Address**: 0x3779c761813D017921481641C16724912978A58F
- **Code Source:**

https://rinkeby.etherscan.io/address/0x3779c761813d017921481641c16724912978a58f#code

Token Information

- Name: OAF
- Total Supply: 500,000,000
- Holders:
- Total transactions:

Contracts address deployed to test net (ETH)

OAF Token smart contract on Eth test net by the auditor to test every function (ETH Test Net)

https://rinkeby.etherscan.io/address/0x886bb2816534da9211a7a5a23848ec75a6f3645b

Executive Summary

According to our assessment, the customer`s solidity smart contract is **Well Secured**.

Well Secured	✓
Secured	
Poor Secured	
Insecure	

Automated checks are with remix IDE. All issues were performed by the team, which included the analysis of code functionality, manual audit found during automated analysis were manually reviewed and applicable vulnerabilities are presented in the audit overview section. The general overview is presented in the Project Information section and all issues found are located in the audit overview section.

Team found 0 critical, 0 high, 0 medium, 2 low, 0 very low-level issues and 1 note in all solidity files of the contract

The files:

OAF.sol

File and Function Level Report

File in Scope:

Contract Name	SHA 256 hash	Contract Address
OAF.sol	**553779ac1ea761e1c08f429f0fd6b2e44b509eddd8f884e33ef10e61718d4d32**	0x886bb2816534da9211a7a5a23848ec75a6f3645b

- Contract: OAF
- Inherit: Context, IERC20, Ownable
- Observation: All passed including security check
- Test Report: passed
- Score: passed
- Conclusion: passed

Function	Test Result	Type / Return Type	Score
name	✓	Read / public	**Passed**
symbol	✓	Read / public	**Passed**
decimals	✓	Read / public	**Passed**
totalSupply	✓	Read / public	**Passed**
allowance	✓	Read / public	**Passed**
balanceOf	✓	Read / public	**Passed**
Owner	✓	Read / public	**Passed**
isExcludedFromFees	✓	Read / public	**Passed**
PlatformFee	✓	Read / public	**Passed**
PlatformWallet	✓	Read / public	**Passed**
totalFees	✓	Read / public	**Passed**
getOwner	✓	Read / public	**Passed**

CommunityPoolFee	✓	Read / public	**Passed**
CommunityPoolWallet	✓	Read / public	**Passed**
burnFee	✓	Read / public	**Passed**
deadWallet	✓	Read / public	**Passed**
divisor	✓	Read / public	**Passed**
devFee	✓	Read / public	**Passed**
devWallet	✓	Read / public	**Passed**
approve	✓	Write / public	**Passed**
TransferFrom	✓	Write / public	**Passed**
increaseAllowance	✓	Write / public	**Passed**
transfer	✓	Write / public	**Passed**
decreaseAllowance	✓	Write / public	**Passed**
includeInFees	✓	Write / public	**Passed**
setBurnFeePercent	✓	Write / public	**Passed**
excludeFromFees	✓	Write / public	**Passed**
setCommunityPoolFeePercent	✓	Write / public	**Passed**
setDevFeePercent	✓	Write / public	**Passed**
renounceOwnership	✓	Write / public	**Passed**
transferOwnership	✓	Write / public	**Passed**
setDevWallet	✓	Write / public	**Passed**
setCommunityPoolWallet	✓	Write / public	**Passed**
setPlatformFeePercent	✓	Write / public	**Passed**
setPlatformWallet	✓	Write / public	**Passed**

Issues Checking Status

No.	Issue Description	Checking Status
1	Compiler warnings.	**Passed**
2	Race conditions and Reentrancy. Cross-function race conditions.	**Passed**
3	Possible delays in data delivery.	**Passed**
4	Oracle calls.	**Passed**
5	Design Logic.	**Passed**
6	Timestamp dependence.	**Passed**
7	Integer Overflow and Underflow.	**Passed**
8	DoS with Revert.	**Passed**
9	DoS with block gas limit.	**Passed with notes**
10	Methods execution permissions.	**Passed**
11	Economy model. If application logic is based on an incorrect economic model, the application would not function correctly and participants would incur financial losses. This type of issue is most often found in bonus rewards systems, Staking and Farming contracts, Vault and Vesting contracts, etc.	**Passed**
12	The impact of the exchange rate on the logic.	**Passed**
13	Private user data leaks.	**Passed**
14	Malicious Event log.	**Passed**
15	Scoping and Declarations.	**Passed**
16	Uninitialized storage pointers.	**Passed**
17	Arithmetic accuracy.	**Passed**

Severity Definitions

Risk Level	Description
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to tokens loss etc.
High	High-level vulnerabilities are difficult to exploit; however, they also have significant impact on smart contract execution, e.g. public access to crucial functions
Medium	Medium-level vulnerabilities are important to fix; however, they can’t lead to tokens lose
Low	Low-level vulnerabilities are mostly related to outdated, unused etc. code snippets, that can’t have significant impact on execution
Note	Lowest-level vulnerabilities, code style violations and info statements can’t affect smart contract execution and can be ignored.

Audit Findings

Critical:

No Critical severity vulnerabilities were found.

High:

No High severity vulnerabilities were found.

Medium:

No Medium severity vulnerabilities were found.

Low:

#Missing zero address validation

Description

When the owner wants to change (dev, community, and platform) wallets, he has to check for the zero address to make, he didn't add the zero address. Otherwise, he will lose the fees.

```
function setPlatformWallet(address _PlatformWallet) external onlyOwner() {
    PlatformWallet = _PlatformWallet;
  }
  function setCommunityPoolWallet(address _CommunityPoolWallet) external
onlyOwner() {
    CommunityPoolWallet = _CommunityPoolWallet;
  }
  function setDevWallet(address _devWallet) external onlyOwner() {
    devWallet = _devWallet;
  }
```

Remediation

Use the require statement to check for zero addresses.

Status: Acknowledged.

#Owner privileges (In the period when the owner isn't renounced)

Description

The owner can change the Fees.
The owner can include / exclude any address from Fees.

```
function setBurnFeePercent(uint256 _burnFee) external onlyOwner() {
    burnFee = _burnFee;
    TotalFee = PlatformFee.add(burnFee).add(CommunityPoolFee).add(devFee);
  }
  function setDevFeePercent(uint256 _devFee) external onlyOwner() {
```

```
    devFee = _devFee;
    TotalFee = PlatformFee.add(burnFee).add(CommunityPoolFee).add(devFee);
  }
  function setPlatformFeePercent(uint256 _PlatformFee) external onlyOwner() {
    PlatformFee = _PlatformFee;
    TotalFee = PlatformFee.add(burnFee).add(CommunityPoolFee).add(devFee);}
  function setCommunityPoolFeePercent(uint256 _CommunityPoolFee) external
onlyOwner() {
    CommunityPoolFee = _CommunityPoolFee;
    TotalFee = PlatformFee.add(burnFee).add(CommunityPoolFee).add(devFee);
  }
function excludeFromFee(address account) public onlyOwner {
    _isExcludedFromFee[account] = true;}
  function includeInFee(address account) public onlyOwner {
    _isExcludedFromFee[account] = false;}
```

Remediation

> Make these functions internal in next version or the team should announce the investors before change the fees and give them time if they want to use the old fees.
>
> P.S: This issue is common to the majority of rewards smart contracts.

Status: Acknowledged.

Very Low:

No Very Low severity vulnerabilities were found.

Notes:

Constant calculations in the contract

Description

recalculated initialization will save 2847 units of gas in deployment

```
    _totalSupply = 500000000 *10**18;
```

Recommendation

Replace the initialization as

```
    _totalSupply = 500000000000000000000000000;
```

Status Acknowledged.

Automatic Testing

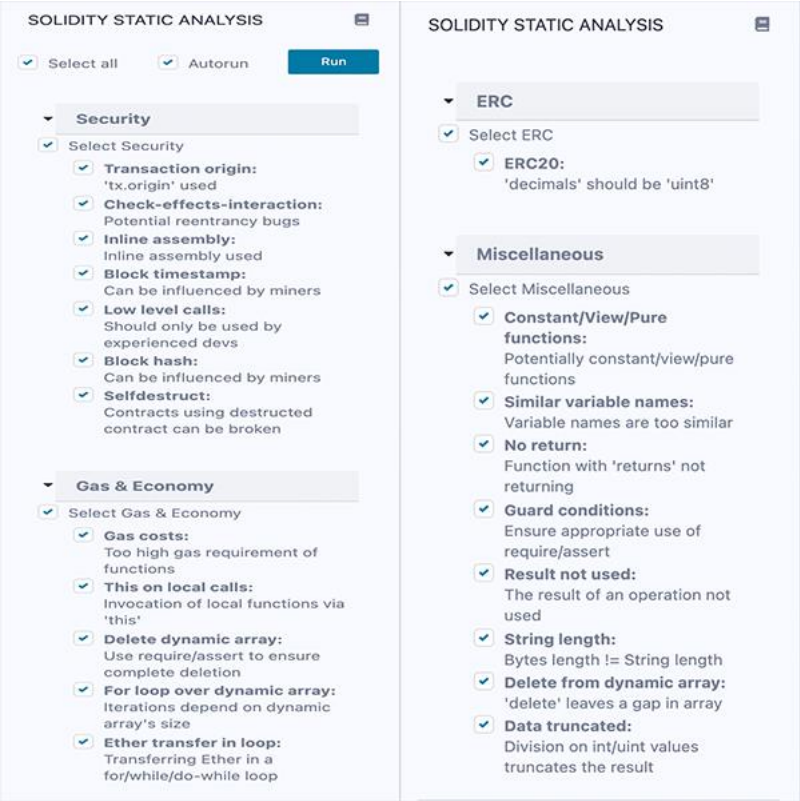
1- Check for security

553779ac1ea761e1c08f429f0fd6b2e44b509eddd8f884e33ef10e61718d4d32

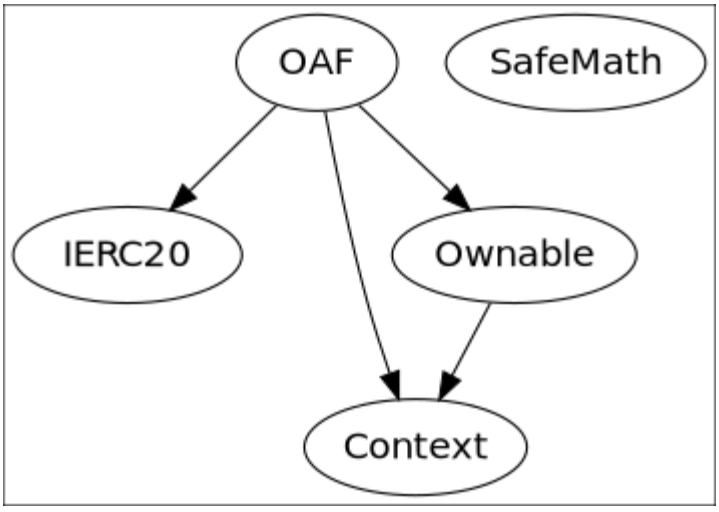
File: OAF.sol | Language: solidity | Size: 21034 bytes | Date: 2022-08-17T09:47:11.417Z

Critical	High	Medium	Low	Note
0	0	0	0	0

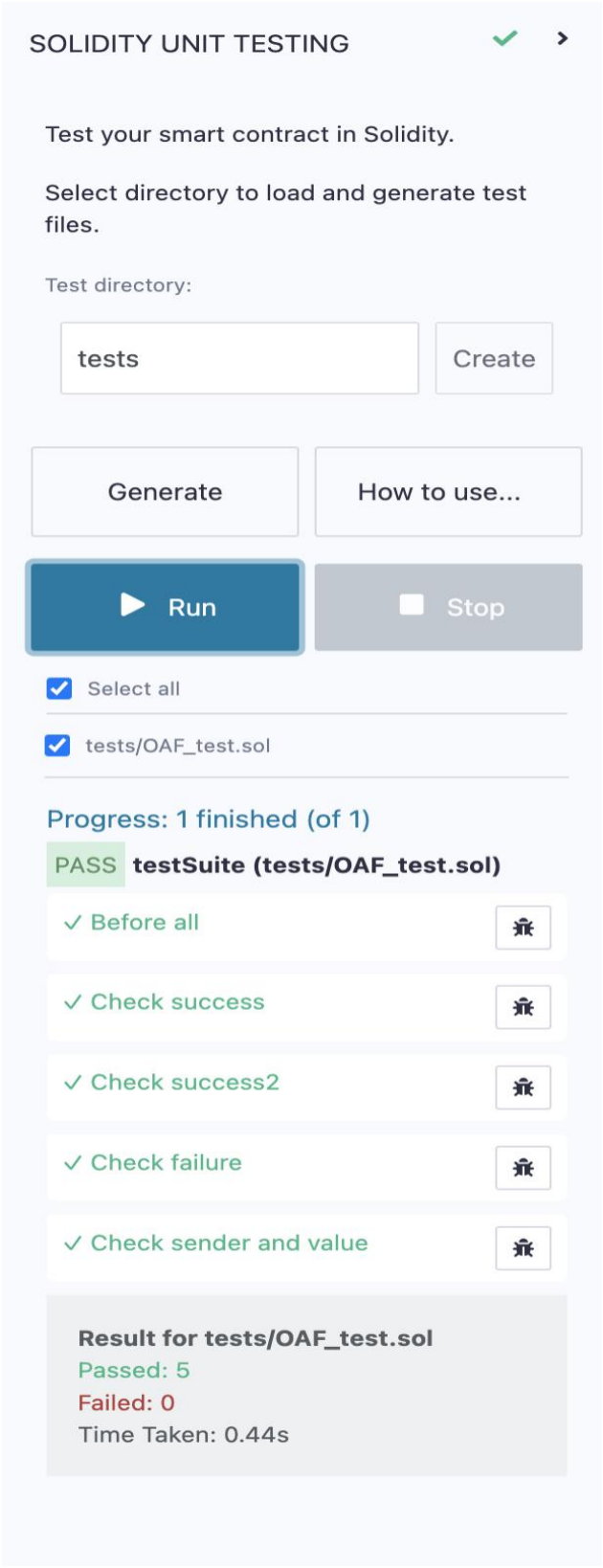
2- SOLIDITY STATIC ANALYSIS



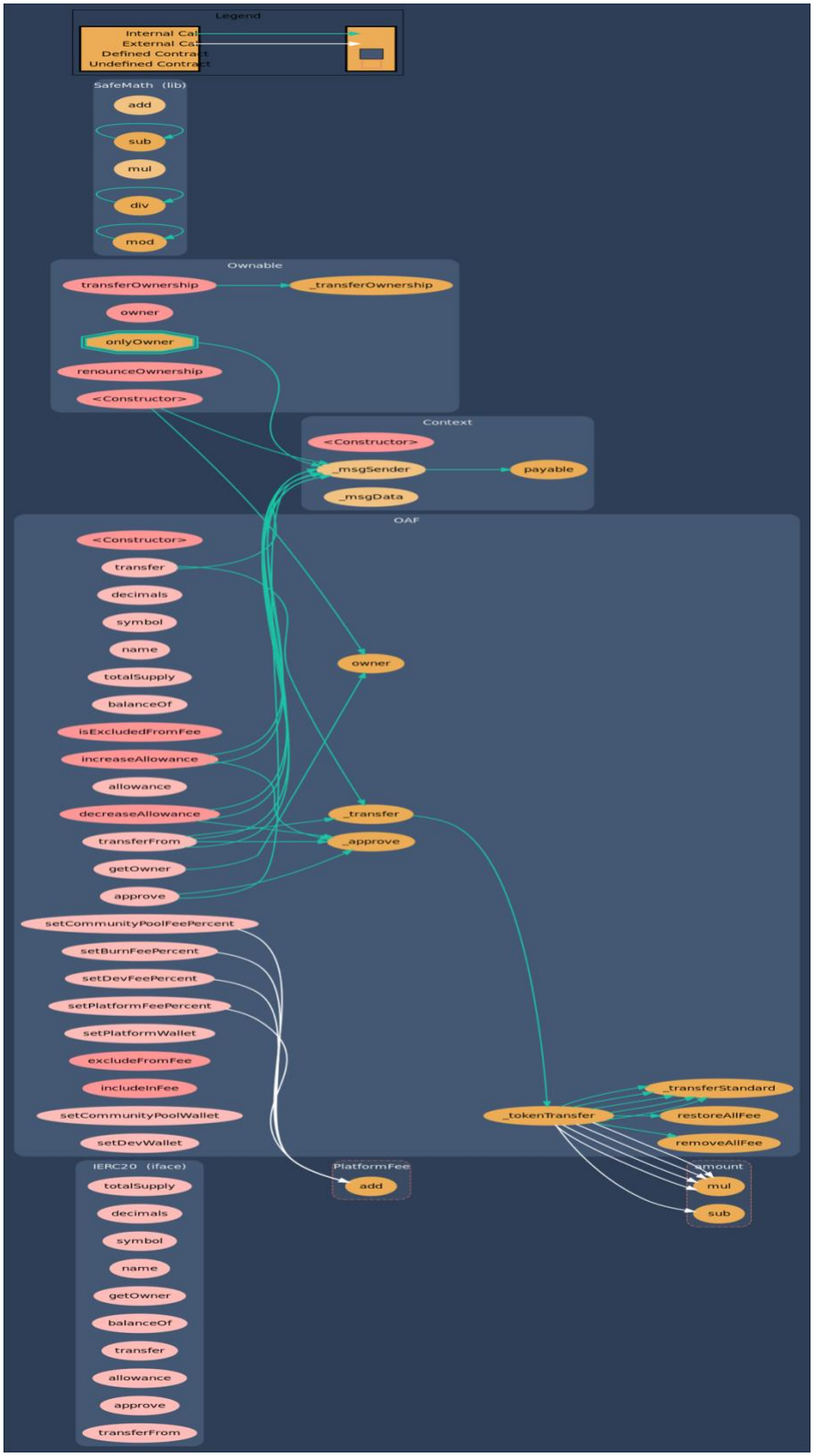
3- Inheritance graph



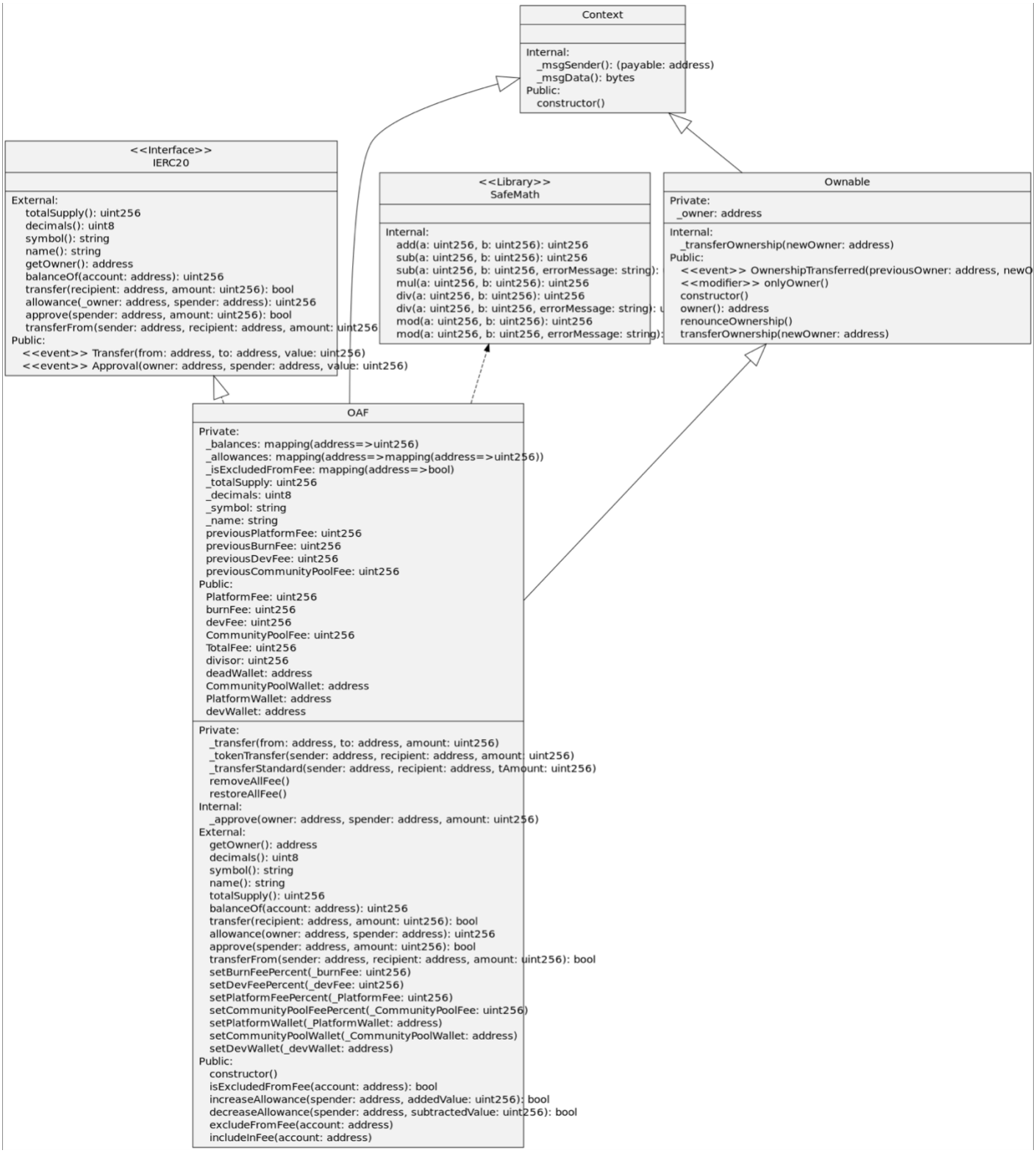
4- SOLIDITY UNIT TESTING



5- Call graph


Legend
Internal Call
External Call
Defined Contract
Undefined Contract
SafeMath (lib)
add
sub
mul
div
mod
Ownable
transferOwnership
_transferOwnership
owner
onlyOwner
renounceOwnership
<Constructor>
Context
<Constructor>
_msgSender
payable
_msgData
OAF
<Constructor>
transfer
decimals
symbol
name
totalSupply
balanceOf
isExcludedFromFee
increaseAllowance
allowance
decreaseAllowance
transferFrom
getOwner
approve
owner
_transfer
_approve
setCommunityPoolFeePercent
setBurnFeePercent
setDevFeePercent
setPlatformFeePercent
setPlatformWallet
excludeFromFee
includeInFee
setCommunityPoolWallet
setDevWallet
_tokenTransfer
_transferStandard
restoreAllFee
removeAllFee
IERC20 (iface)
totalSupply
decimals
symbol
name
getOwner
balanceOf
transfer
allowance
approve
transferFrom
PlatformFee
add
amount
mul
sub

Unified Modeling Language (UML)



Functions signature

```
Sighash   |   Function Signature
========================
39509351  =>  increaseAllowance(address,uint256)
18160ddd  =>  totalSupply()
313ce567  =>  decimals()
95d89b41  =>  symbol()
06fdde03  =>  name()
893d20e8  =>  getOwner()
70a08231  =>  balanceOf(address)
a9059cbb  =>  transfer(address,uint256)
dd62ed3e  =>  allowance(address,address)
095ea7b3  =>  approve(address,uint256)
23b872dd  =>  transferFrom(address,address,uint256)
119df25f  =>  _msgSender()
8b49d47e  =>  _msgData()
771602f7  =>  add(uint256,uint256)
b67d77c5  =>  sub(uint256,uint256)
e31bdc0a  =>  sub(uint256,uint256,string)
c8a4ac9c  =>  mul(uint256,uint256)
a391c15b  =>  div(uint256,uint256)
b745d336  =>  div(uint256,uint256,string)
f43f523a  =>  mod(uint256,uint256)
71af23e8  =>  mod(uint256,uint256,string)
8da5cb5b  =>  owner()
715018a6  =>  renounceOwnership()
f2fde38b  =>  transferOwnership(address)
d29d44ee  =>  _transferOwnership(address)
5342acb4  =>  isExcludedFromFee(address)
a457c2d7  =>  decreaseAllowance(address,uint256)
30e0789e  =>  _transfer(address,address,uint256)
f147aa74  =>  _tokenTransfer(address,address,uint256)
2852df65  =>  _transferStandard(address,address,uint256)
301370af  =>  removeAllFee()
e7e3e3a7  =>  restoreAllFee()
104e81ff  =>  _approve(address,address,uint256)
cea26958  =>  setBurnFeePercent(uint256)
379e2919  =>  setDevFeePercent(uint256)
cab230d2  =>  setPlatformFeePercent(uint256)
79e534e9  =>  setCommunityPoolFeePercent(uint256)
8831e9cf  =>  setPlatformWallet(address)
437823ec  =>  excludeFromFee(address)
ea2f0b37  =>  includeInFee(address)
6b79c97f  =>  setCommunityPoolWallet(address)
1f53ac02  =>  setDevWallet(address)
```

Automatic general report

Files Description Table

File Name	SHA-1 Hash
/Users/macbook/Desktop/smart contracts/OAF.sol	589ed2022a140297aa178b419b8c60057115841a

Contracts Description Table

Contract	Type	Bases		
└	**Function Name**	**Visibility**	**Mutability**	**Modifiers**
IERC20	Interface			
└	totalSupply	External ❗		NO❗
└	decimals	External ❗		NO❗
└	symbol	External ❗		NO❗
└	name	External ❗		NO❗
└	getOwner	External ❗		NO❗
└	balanceOf	External ❗		NO❗
└	transfer	External ❗	🛑	NO❗
└	allowance	External ❗		NO❗
└	approve	External ❗	🛑	NO❗
└	transferFrom	External ❗	🛑	NO❗
Context	Implementation			
└	<Constructor>	Public ❗	🛑	NO❗
└	_msgSender	Internal 🔒		
└	_msgData	Internal 🔒		
SafeMath	Library			
└	add	Internal 🔒		
└	sub	Internal 🔒		
└	sub	Internal 🔒		
└	mul	Internal 🔒		
└	div	Internal 🔒		
└	div	Internal 🔒		
└	mod	Internal 🔒		
└	mod	Internal 🔒		
Ownable	Implementation	Context		
└	<Constructor>	Public ❗	🛑	NO❗
└	owner	Public ❗		NO❗
└	renounceOwnership	Public ❗	🛑	onlyOwner
└	transferOwnership	Public ❗	🛑	onlyOwner
└	_transferOwnership	Internal 🔒	🛑	
OAF	Implementation	Context, IERC20, Ownable		
└	<Constructor>	Public ❗	🛑	NO❗
└	getOwner	External ❗		NO❗
└	decimals	External ❗		NO❗
└	symbol	External ❗		NO❗

└	name	External ❗		NO❗
└	totalSupply	External ❗		NO❗
└	balanceOf	External ❗		NO❗
└	isExcludedFromFee	Public ❗		NO❗
└	transfer	External ❗	🛑	NO❗
└	allowance	External ❗		NO❗
└	approve	External ❗	🛑	NO❗
└	transferFrom	External ❗	🛑	NO❗
└	increaseAllowance	Public ❗	🛑	NO❗
└	decreaseAllowance	Public ❗	🛑	NO❗
└	_transfer	Private 🔐	🛑	
└	_tokenTransfer	Private 🔐	🛑	
└	_transferStandard	Private 🔐	🛑	
└	removeAllFee	Private 🔐	🛑	
└	restoreAllFee	Private 🔐	🛑	
└	_approve	Internal 🔒	🛑	
└	setBurnFeePercent	External ❗	🛑	onlyOwner
└	setDevFeePercent	External ❗	🛑	onlyOwner
└	setPlatformFeePercent	External ❗	🛑	onlyOwner
└	setCommunityPoolFeePercent	External ❗	🛑	onlyOwner
└	setPlatformWallet	External ❗	🛑	onlyOwner
└	excludeFromFee	Public ❗	🛑	onlyOwner
└	includeInFee	Public ❗	🛑	onlyOwner
└	setCommunityPoolWallet	External ❗	🛑	onlyOwner
└	setDevWallet	External ❗	🛑	onlyOwner

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

Conclusion

The contracts are written systematically. Team found no critical issues. So, it is good to go for production.

Since possible test cases can be unlimited and developer level documentation (code flow diagram with function level description) not provided, for such an extensive smart contract protocol, we provide no such guarantee of future outcomes. We have used all the latest static tools and manual observations to cover maximum possible test cases to scan Everything.

Security state of the reviewed contract is “Well Secured”.

✓ No mint function.
✓ No volatile code.
✓ No high severity issues were found.

Disclaimer

This is a limited report on our findings based on our analysis, in accordance with good industry practice as of the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, the details of which are set out in this report. In order to get a full view of our analysis, it is crucial for you to read the full report. While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against the team on the basis of what it says or doesn't say, or how team produced it, and it is important for you to conduct your own independent investigations before making any decisions. team go into more detail on this in the below disclaimer below – please make sure to read it in full.

By reading this report or any part of it, you agree to the terms of this disclaimer. If you do not agree to the terms, then please immediately cease reading this report, and delete and destroy any and all copies of this report downloaded and/or printed by you. This report is provided for information purposes only and on a non-reliance basis, and does not constitute investment advice. No one shall have any right to rely on the report or its contents, and Saferico and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) (Saferico s) owe no duty of care towards you or any other person, nor does Saferico make any warranty or representation to any person on the accuracy or completeness of the report. The report is provided "as is", without any conditions, warranties or other terms of any kind except as set out in this disclaimer, and Saferico hereby excludes all representations, warranties, conditions and other terms (including, without limitation, the warranties implied by law of satisfactory quality, fitness for purpose and the use of reasonable care and skill) which, but for this clause, might have effect in relation to the report. Except and only to the extent that it is prohibited by law, Saferico hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against Saferico, for any amount or kind of loss or damage that may result to you or any other person (including without limitation, any direct, indirect, special, punitive, consequential or pure economic loss or damages, or any loss of income, profits, goodwill, data, contracts, use of money, or business interruption, and whether in delict, tort (including without limitation negligence), contract, breach of statutory duty, misrepresentation (whether innocent or negligent) or otherwise under any claim of any nature whatsoever in any jurisdiction) in any way arising from or connected with this report and the use, inability to use or the results of use of this report, and any reliance on this report. The analysis of the security is purely based on the smart contracts alone. No applications or operations were reviewed for security. No product code has been reviewed.