Smart Contract Security Audit V1

Digital UYS Crowd Sale

21/2/2022

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**: 0xda3A87B881EA63076c05bD647cC7fB0F66ad53f5
- **Code:**

https://rinkeby.etherscan.io/address/0xda3a87b881ea63076c05bd647cc7fb0f66ad53f5#code

Contracts address deployed to test net (ETH)

Digital UYS Crowd Sale Smart contract on ETH test net to test write functions by the auditor.

https://rinkeby.etherscan.io/address/0xcf9f1fcb629e216002b26b6be403ca5f136e350c

https://rinkeby.etherscan.io/address/0xd0ada8eec055816cc115f91daa03351889321f8b

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, 1 low, 0 very low-level issues and 1 note in all solidity files of the contract

The files:

CrowdsaleDigitalUYS.sol

File and Function Level Report

File in Scope:

Contract Name	SHA 256 hash	Contract Address
CrowdsaleDigitalUYS.sol	**4ef42070ad347ae3ff6bfed45db4d0db77b6849d3daf7d8e3fda09358b731551**	0xda3A87B881EA63076c05bD647cC7fB0F66ad53f5

- Contract: CrowdsaleDigitalUYS
- Inherit: Context
- Observation: All passed including security check
- Test Report: passed
- Score: passed
- Conclusion: passed

Function	Test Result	Type / Return Type	Score
rate	✓	Read / public	**Passed**
remainingTokens	✓	Read / public	**Passed**
token	✓	Read / public	**Passed**
totalETHCollected	✓	Read / public	**Passed**
wallet	✓	Read / public	**Passed**
weiRaised	✓	Read / public	**Passed**
buyTokens	✓	Write / payable	**Passed**
changeRate	✓	Write / public	**Passed**
endIco	✓	Write / public	**Passed**
withdraw	✓	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	Timestamp dependence.	**Passed**
6	Integer Overflow and Underflow.	**Passed**
7	DoS with Revert.	**Passed**
8	DoS with block gas limit.	**Passed with notes**
9	Methods execution permissions.	**Passed**
10	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**
11	The impact of the exchange rate on the logic.	**Passed**
12	Private user data leaks.	**Passed**
13	Malicious Event log.	**Passed**
14	Scoping and Declarations.	**Passed**
15	Uninitialized storage pointers.	**Passed**
16	Arithmetic accuracy.	**Passed**
17	Design Logic.	**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:

No low severity vulnerabilities were found

Very Low:

No Very Low severity vulnerabilities were found.

Notes:

#Compiler version is old
Description
The compiler being used was released 2-3 years ago. It's recommended to use more recent compiler version, there can be benefits like reduction in bytecode size etc.

Status: Acknowledged

Automatic Testing

1- Check for security

4ef42070ad347ae3ff6bfed45db4d0db77b6849d3daf7d8e3fda09358b731551

File: CrowdS... | Language: solidity | Size: 8228 bytes | Date: 2022-02-21T09:26:56.069Z

Critical	High	Medium	Low	Note
0	0	0	0	0

2- SOLIDITY STATIC ANALYSIS

SOLIDITY STATIC ANALYSIS

Select all | Autorun | Run

Security

Select Security

- **Transaction origin:** 'tx.origin' used
- **Check-effects-interaction:** Potential reentrancy bugs
- **Inline assembly:** Inline assembly used
- **Block timestamp:** Can be influenced by miners
- **Low level calls:** Should only be used by experienced devs
- **Block hash:** Can be influenced by miners
- **Selfdestruct:** Contracts using destructed contract can be broken

Gas & Economy

Select Gas & Economy

- **Gas costs:** Too high gas requirement of functions
- **This on local calls:** Invocation of local functions via 'this'
- **Delete dynamic array:** Use require/assert to ensure complete deletion
- **For loop over dynamic array:** Iterations depend on dynamic array's size
- **Ether transfer in loop:** Transferring Ether in a for/while/do-while loop

SOLIDITY STATIC ANALYSIS

ERC

Select ERC

- **ERC20:** 'decimals' should be 'uint8'

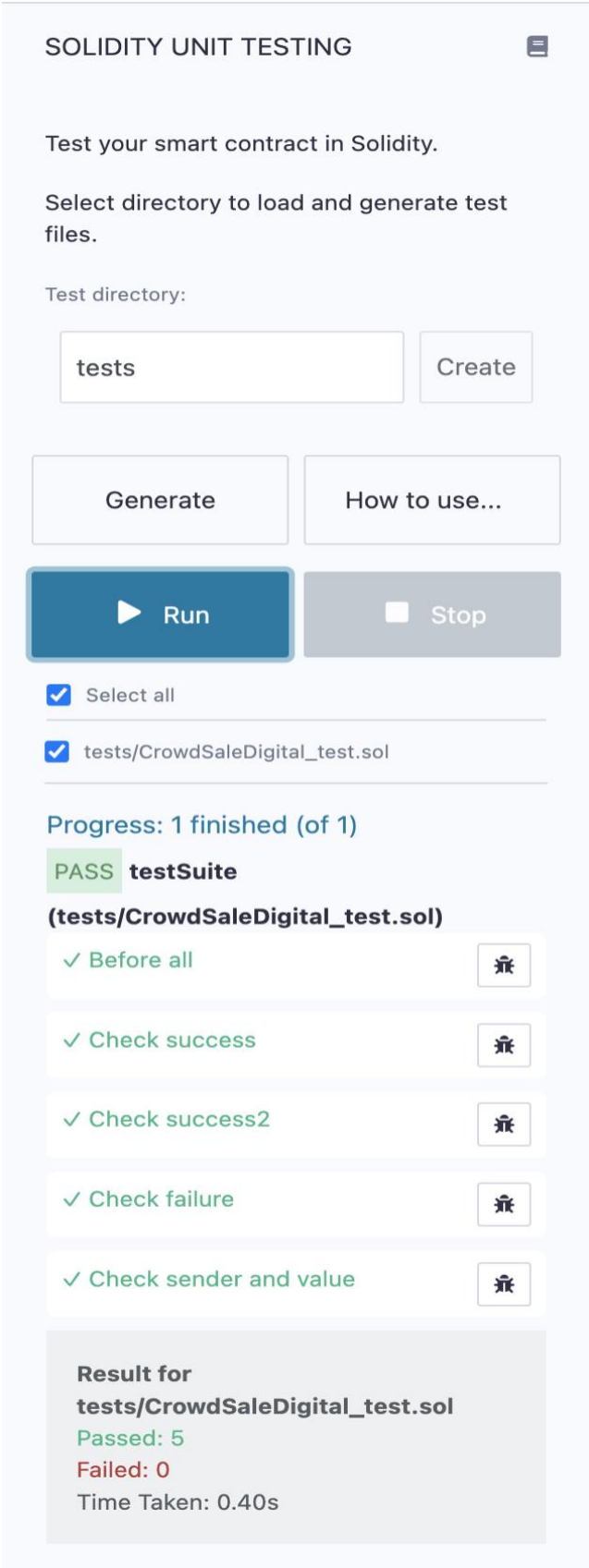
Miscellaneous

Select Miscellaneous

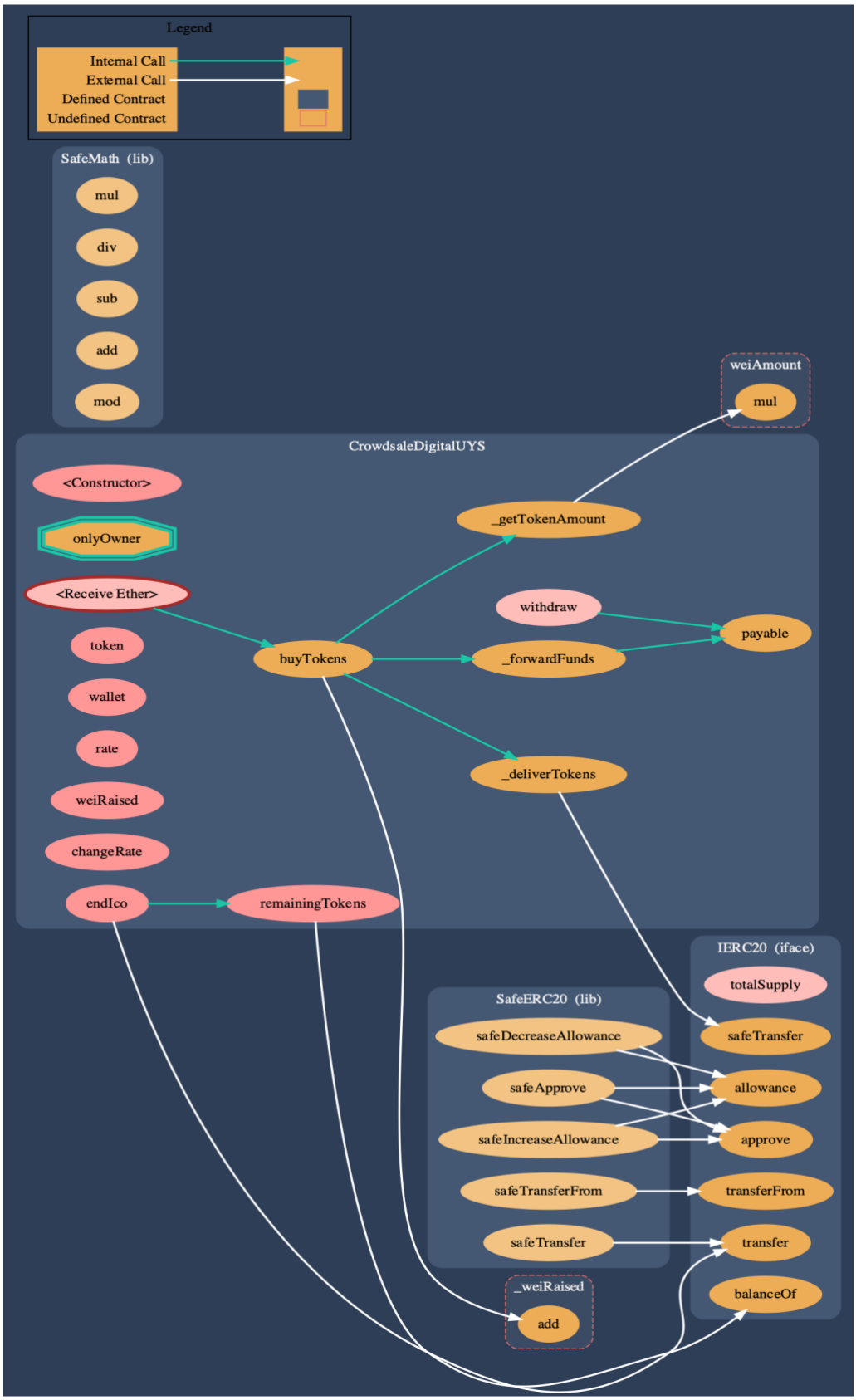
- **Constant/View/Pure functions:** Potentially constant/view/pure functions
- **Similar variable names:** Variable names are too similar
- **No return:** Function with 'returns' not returning
- **Guard conditions:** Ensure appropriate use of require/assert
- **Result not used:** The result of an operation not used
- **String length:** Bytes length != String length
- **Delete from dynamic array:** 'delete' leaves a gap in array
- **Data truncated:** Division on int/uint values truncates the result

3- Inheritance graph

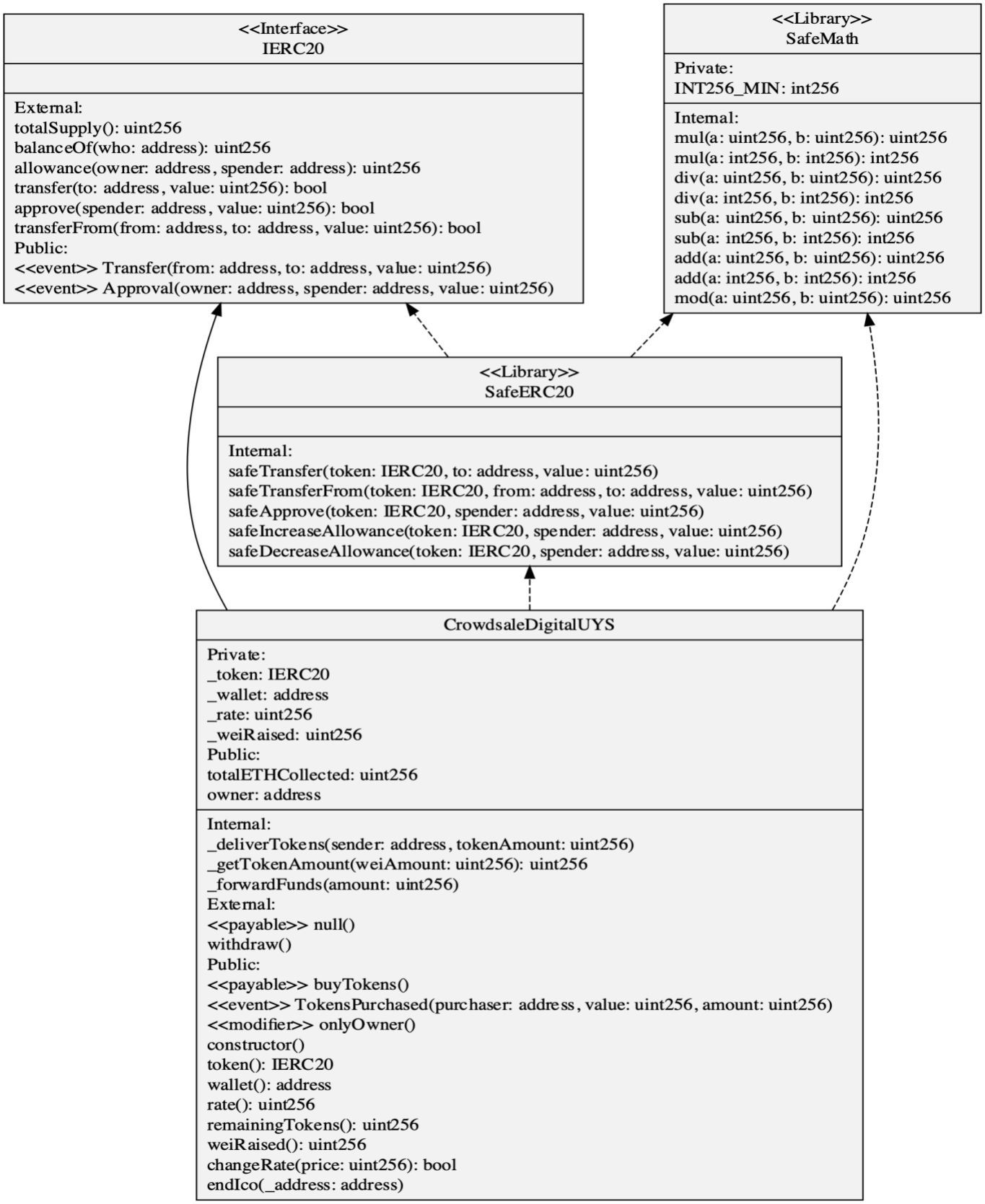
4- SOLIDITY UNIT TESTING



5- Call graph


Legend
Internal Call
External Call
Defined Contract
Undefined Contract
SafeMath (lib)
mul
div
sub
add
mod
weiAmount
mul
CrowdsaleDigitalUYS
<Constructor>
onlyOwner
<Receive Ether>
token
wallet
rate
weiRaised
changeRate
endIco
remainingTokens
buyTokens
_getTokenAmount
withdraw
_forwardFunds
payable
_deliverTokens
IERC20 (iface)
totalSupply
safeTransfer
allowance
approve
transferFrom
transfer
balanceOf
SafeERC20 (lib)
safeDecreaseAllowance
safeApprove
safeIncreaseAllowance
safeTransferFrom
safeTransfer
_weiRaised
add

Unified Modeling Language (UML)



Functions signature

```
43509138  =>  div(int256,int256)
18160ddd  =>  totalSupply()
70a08231  =>  balanceOf(address)
dd62ed3e  =>  allowance(address,address)
a9059cbb  =>  transfer(address,uint256)
095ea7b3  =>  approve(address,uint256)
23b872dd  =>  transferFrom(address,address,uint256)
c8a4ac9c  =>  mul(uint256,uint256)
bbe93d91  =>  mul(int256,int256)
a391c15b  =>  div(uint256,uint256)
b67d77c5  =>  sub(uint256,uint256)
adefc37b  =>  sub(int256,int256)
771602f7  =>  add(uint256,uint256)
a5f3c23b  =>  add(int256,int256)
f43f523a  =>  mod(uint256,uint256)
d0c407e1  =>  safeTransfer(IERC20,address,uint256)
5beae096  =>  safeTransferFrom(IERC20,address,address,uint256)
d6dcec8d  =>  safeApprove(IERC20,address,uint256)
390cc046  =>  safeIncreaseAllowance(IERC20,address,uint256)
5164ffed  =>  safeDecreaseAllowance(IERC20,address,uint256)
fc0c546a  =>  token()
521eb273  =>  wallet()
2c4e722e  =>  rate()
bf583903  =>  remainingTokens()
4042b66f  =>  weiRaised()
74e7493b  =>  changeRate(uint256)
d0febe4c  =>  buyTokens()
ed2cbf06  =>  _deliverTokens(address,uint256)
7a99bb0a  =>  _getTokenAmount(uint256)
b3413d9f  =>  _forwardFunds()
0339d81c  =>  endIco(address)
3ccfd60b  =>  withdraw()
```

Automatic general report

```
Files Description Table

|  File Name  |  SHA-1 Hash  |
|-------------|--------------|
| /Users/macbook/Desktop/smart contracts/CrowdSaleDigital.sol |
90d3e829504df0d63d82645d941eed9d3ab65d41 |


 Contracts Description Table

|  Contract  |         Type        |       Bases      |
|                  |
|:----------:|:-------------------:|:----------------:|:----------------
:|:---------------:|
|      └      |  **Function Name**  |  **Visibility**  |  **Mutability**
|  **Modifiers**  |
||||||
| **IERC20** | Interface |  |||
| └ | totalSupply | External ❗ |   |NO❗ |
| └ | balanceOf | External ❗ |   |NO❗ |
| └ | allowance | External ❗ |   |NO❗ |
| └ | transfer | External ❗ | 🛑  |NO❗ |
| └ | approve | External ❗ | 🛑  |NO❗ |
| └ | transferFrom | External ❗ | 🛑  |NO❗ |
||||||
| **SafeMath** | Library |  |||
| └ | mul | Internal 🔒 |   | |
| └ | mul | Internal 🔒 |   | |
| └ | div | Internal 🔒 |   | |
| └ | div | Internal 🔒 |   | |
| └ | sub | Internal 🔒 |   | |
| └ | sub | Internal 🔒 |   | |
| └ | add | Internal 🔒 |   | |
| └ | add | Internal 🔒 |   | |
| └ | mod | Internal 🔒 |   | |
||||||
| **SafeERC20** | Library |  |||
| └ | safeTransfer | Internal 🔒 | 🛑  | |
| └ | safeTransferFrom | Internal 🔒 | 🛑  | |
| └ | safeApprove | Internal 🔒 | 🛑  | |
| └ | safeIncreaseAllowance | Internal 🔒 | 🛑  | |
| └ | safeDecreaseAllowance | Internal 🔒 | 🛑  | |
||||||
| **Crowdsale** | Implementation |  |||
| └ | <Constructor> | Public ❗ | 🛑  |NO❗ |
| └ | <Receive Ether> | External ❗ |  💶 |NO❗ |
```

└	token	Public ❗		NO❗
└	wallet	Public ❗		NO❗
└	rate	Public ❗		NO❗
└	remainingTokens	Public ❗		NO❗
└	weiRaised	Public ❗		NO❗
└	changeRate	Public ❗	🛑	onlyOwner
└	buyTokens	Public ❗	💵	NO❗
└	_deliverTokens	Internal 🔒	🛑	
└	_getTokenAmount	Internal 🔒		
└	_forwardFunds	Internal 🔒	🛑	
└	endIco	Public ❗	🛑	onlyOwner

Contracts Description Table

Contract	Type	Bases

└	**Function Name**	**Visibility**	**Mutability**	**Modifiers**
IERC20	Interface				
└	totalSupply	External ❗		NO❗
└	balanceOf	External ❗		NO❗
└	allowance	External ❗		NO❗
└	transfer	External ❗	🛑	NO❗
└	approve	External ❗	🛑	NO❗
└	transferFrom	External ❗	🛑	NO❗
SafeMath	Library				
└	mul	Internal 🔒		
└	mul	Internal 🔒		
└	div	Internal 🔒		
└	div	Internal 🔒		
└	sub	Internal 🔒		
└	sub	Internal 🔒		
└	add	Internal 🔒		
└	add	Internal 🔒		
└	mod	Internal 🔒		
SafeERC20	Library				
└	safeTransfer	Internal 🔒	🛑	
└	safeTransferFrom	Internal 🔒	🛑	
└	safeApprove	Internal 🔒	🛑	
└	safeIncreaseAllowance	Internal 🔒	🛑	
└	safeDecreaseAllowance	Internal 🔒	🛑	
Crowdsale	Implementation				

└	<Constructor>	Public ❗	🛑	NO❗	
└	<Receive Ether>	External ❗		💵	NO❗
└	token	Public ❗		NO❗	
└	wallet	Public ❗		NO❗	
└	rate	Public ❗		NO❗	
└	remainingTokens	Public ❗		NO❗	
└	weiRaised	Public ❗		NO❗	
└	changeRate	Public ❗	🛑		onlyOwner
└	buyTokens	Public ❗		💵	NO❗
└	_deliverTokens	Internal 🔒	🛑		
└	_getTokenAmount	Internal 🔒			
└	_forwardFunds	Internal 🔒	🛑		
└	endIco	Public ❗	🛑		onlyOwner
└	withdraw	External ❗	🛑		onlyOwner

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

Conclusion

The contracts are written systematically. Team found no critical. 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 volatile code.
✓ Not many 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.