Smart Contract
Security Audit
V1

Edbit Smart Contract

25/2/2022

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**: 0x2428Ec4059189Ea652C55126d03279991F54822B
- **Code:**

https://github.com/edbitio/edt-smartcontract/blob/main/edbit-edt-erc20.sol

Contracts address deployed to test net (Ethereum)

Edbit Smart contract on ETH test net to test every function by the auditor.

https://rinkeby.etherscan.io/address/0x2428ec4059189ea652c55126d03279991f54822b

Executive Summary

According to our assessment, the customer`s solidity smart contract is **Well-Secured**. Because the team fix all high and low issues.

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

The files:

Edbit smart contract.sol

File and Function Level Report

File in Scope:

Contract Name	SHA 256 hash	Contract Address
Edbit.sol	**05d12d9df191b0aef38f3794ba9460adbce0598b4de8806a483a1404aba6b8d1**	0x2428Ec4059189Ea652C55126d03279991F54822B

- Contract: Edbit
- Inherit: ERC20Decimals, ERC20Capped, ERC20Mintable, ERC20Burnable, ERC1363, TokenRecover, Roles
- 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**
cap	✓	Read / public	**Passed**
supportsInterface	✓	Read / public	**Passed**
totalSupply	✓	Read / public	**Passed**
balanceOf	✓	Read / public	**Passed**
Owner	✓	Read / public	**Passed**
decimal	✓	Read / public	**Passed**
getRoleAdmin	✓	Read / public	**Passed**
MINTER_ROLE	✓	Read / public	**Passed**
hasRole	✓	Read / public	**Passed**
mintingFinished	✓	Read / public	**Passed**

allowance	✓	Read / public	Passed
DEFAULT_ADMIN_ROLE	✓	Read / public	Passed
burn	✓	Write / public	Passed
approve	✓	Write / public	Passed
burnFrom	✓	Write / public	Passed
approveAndCall	✓	Write / public	Passed
approveAndCall	✓	Write / public	Passed
decreaseAllowance	✓	Write / public	Passed
mint	✓	Write / public	Passed
increaseAllowance	✓	Write / public	Passed
transferOwnership	✓	Write / public	Passed
finishMinting	✓	Write / public	Passed
transferFrom	✓	Write / public	Passed
transfer	✓	Write / public	Passed
recoverERC20	✓	Write / public	Passed
renounceOwnership	✓	Write / public	Passed
mint	✓	Write / public	Passed
renounceRole	✓	Write / public	Passed
revokeRole	✓	Write / public	Passed
transferFromAndCall	✓	Write / public	Passed
transferFromAndCall	✓	Write / public	Passed
transferAndCall	✓	Write / public	Passed
transferAndCall	✓	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:

#Contract code size exceeds 24576 bytes

Description

Contract implementation is too large in size to be deployed on mainnet. Ethereum with its spurious dragon release limited the size of the contracts deployable on mainnet to 24576 bytes.
The size of the contract Edbit.sol goes way above this value and currently is of size 59057bytes.

Remediation

Define and use libraries for pure and view functions e.g. We can create a library which contains all the mathematical operations.

Status: Closed. Fixed in version 2.

Medium:

No Medium severity vulnerabilities were found

Low:

#Multiple pragma statements

Line	Pragma
20	pragma solidity ^0.8.0;
100	pragma solidity ^0.8.0;
129	pragma solidity ^0.8.0;
156	pragma solidity ^0.8.0;
462	pragma solidity ^0.8.0;
504	pragma solidity ^0.8.0;
542	pragma solidity ^0.8.0;
734	pragma solidity ^0.8.0;

761	pragma solidity ^0.8.0;
791	pragma solidity ^0.8.0;
868	pragma solidity ^0.8.0;
898	pragma solidity ^0.8.0;
927	pragma solidity ^0.8.0;
1062	pragma solidity ^0.8.0;
1132	pragma solidity ^0.8.0;
1156	pragma solidity ^0.8.0;
1183	pragma solidity ^0.8.0;
1250	pragma solidity ^0.8.0;
1320	pragma solidity ^0.8.0;
1565	pragma solidity ^0.8.0;
1587	pragma solidity ^0.8.0;
1608	pragma solidity ^0.8.0;

Description
There are multiple pragma statements in the code. Only the compiler version 0.8.2 will work with the code, but keeping only one pragma statement helps in maintaining readability of the code.

Remediation

Keep a single pragma statement.

Status: Closed. Fixed In version 2

Very Low:

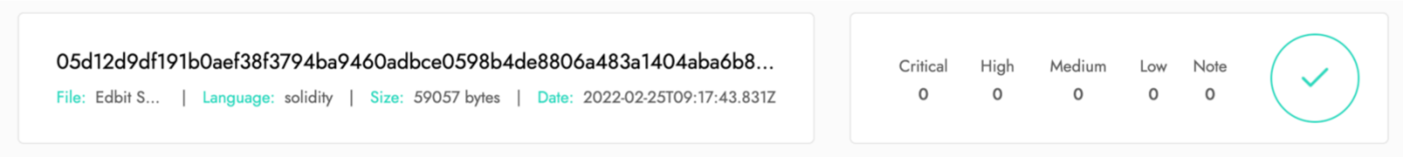
No Very Low severity vulnerabilities were found.

Notes:

No Notes were found

Automatic Testing

1- Check for security



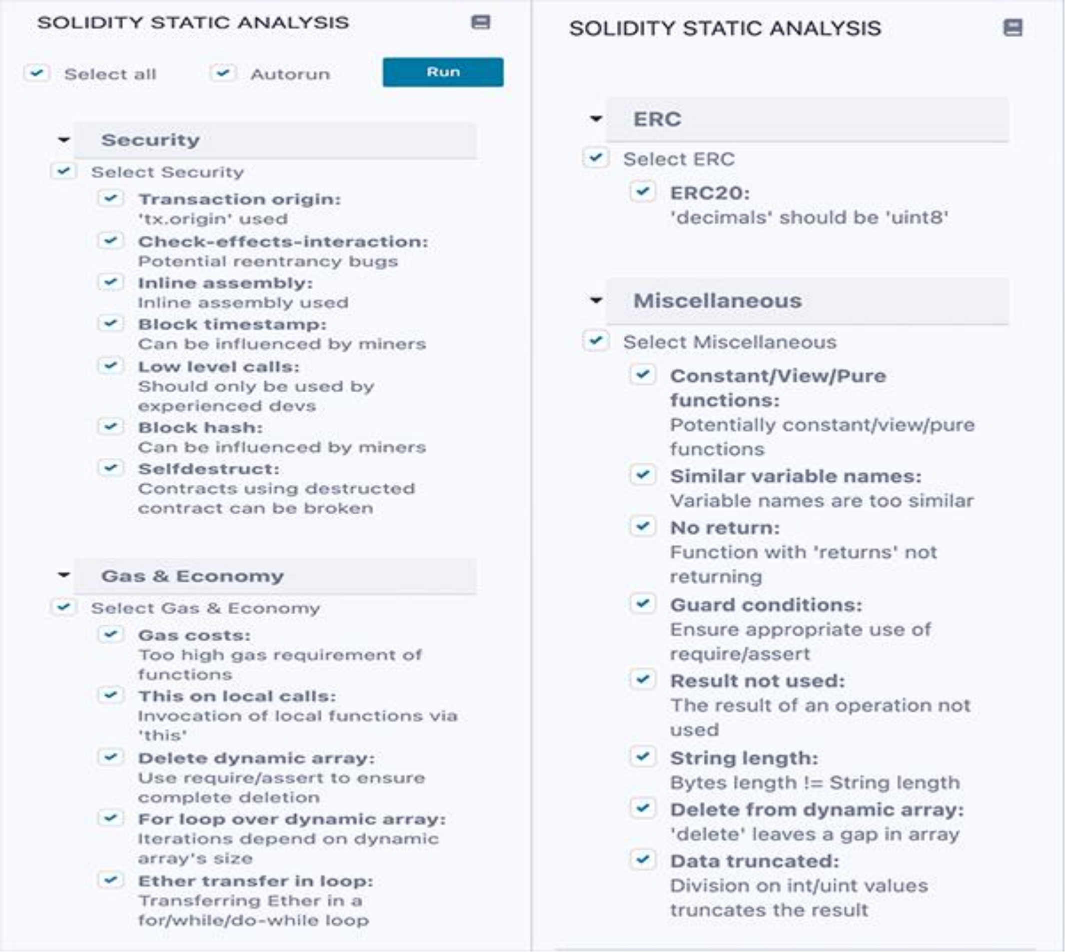
05d12d9df191b0aef38f3794ba9460adbce0598b4de8806a483a1404aba6b8...

File: Edbit S... | Language: solidity | Size: 59057 bytes | Date: 2022-02-25T09:17:43.831Z

Critical	High	Medium	Low	Note
0	0	0	0	0

✓

2- SOLIDITY STATIC ANALYSIS



SOLIDITY STATIC ANALYSIS

☑ Select all ☑ Autorun

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

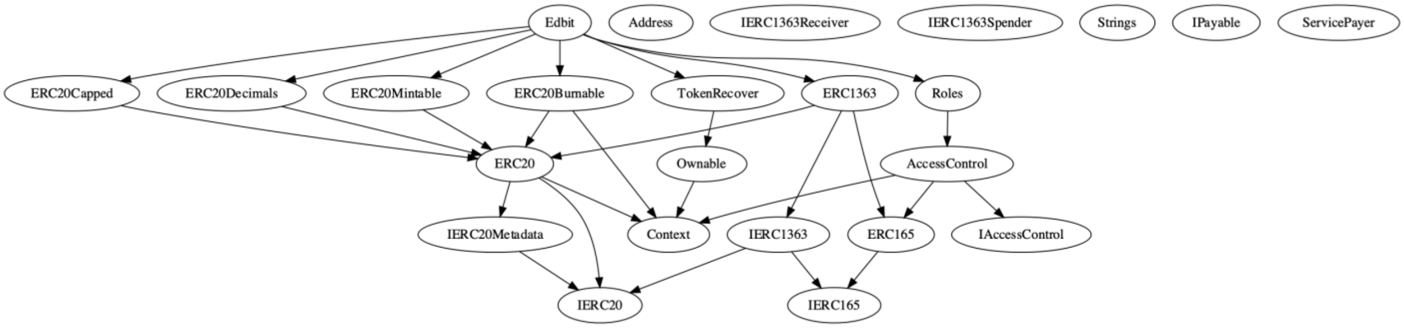
- ☑ **ERC20:** 'decimals' should be 'uint8'

Miscellaneous

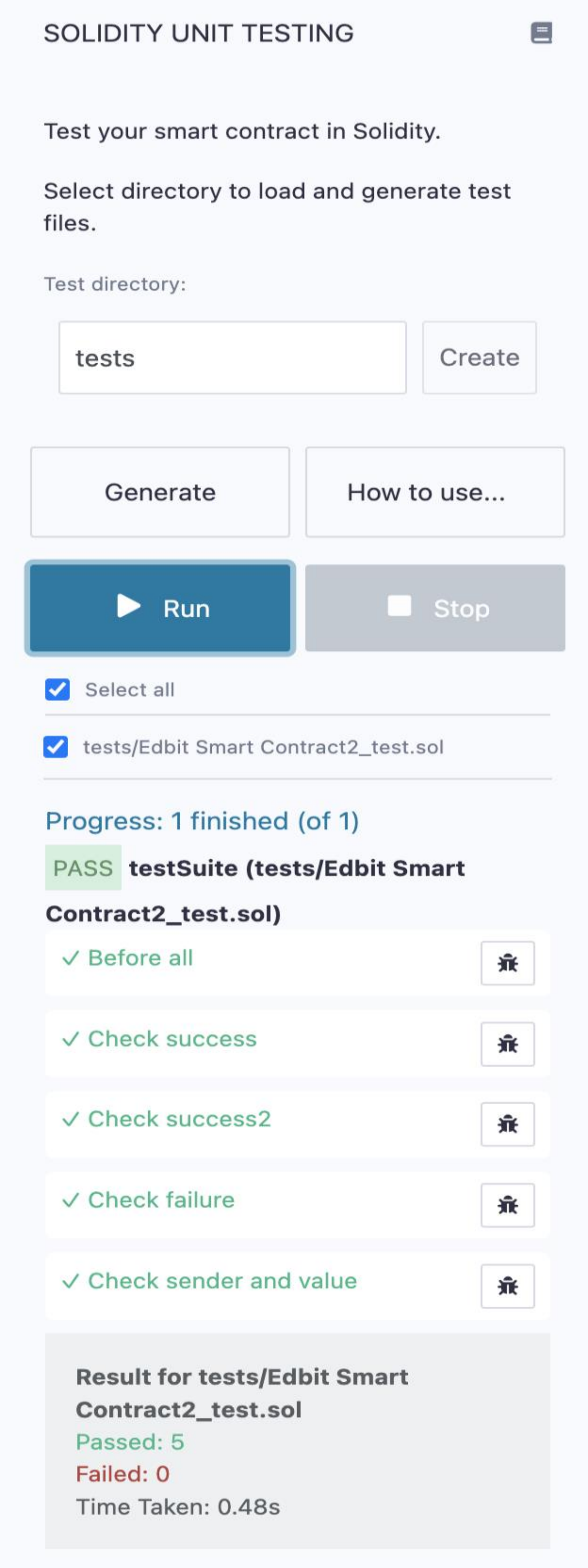
☑ Select Miscellaneous

- ☑ **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



Test your smart contract in Solidity.

Select directory to load and generate test files.

Test directory: tests

☑ Select all

☑ tests/Edbit Smart Contract2_test.sol

Progress: 1 finished (of 1)

PASS **testSuite (tests/Edbit Smart Contract2_test.sol)**

✓ Before all

✓ Check success

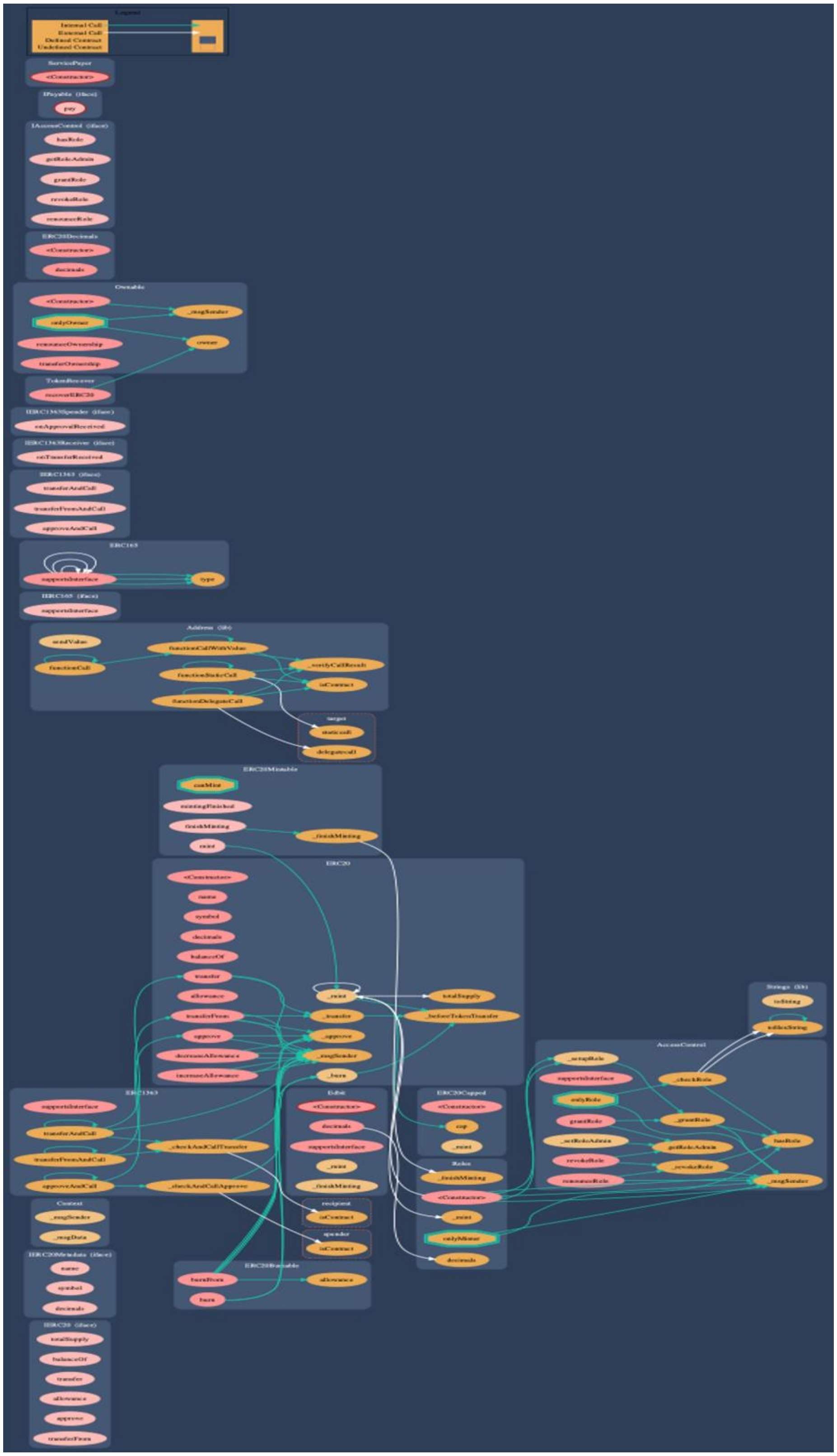
✓ Check success2

✓ Check failure

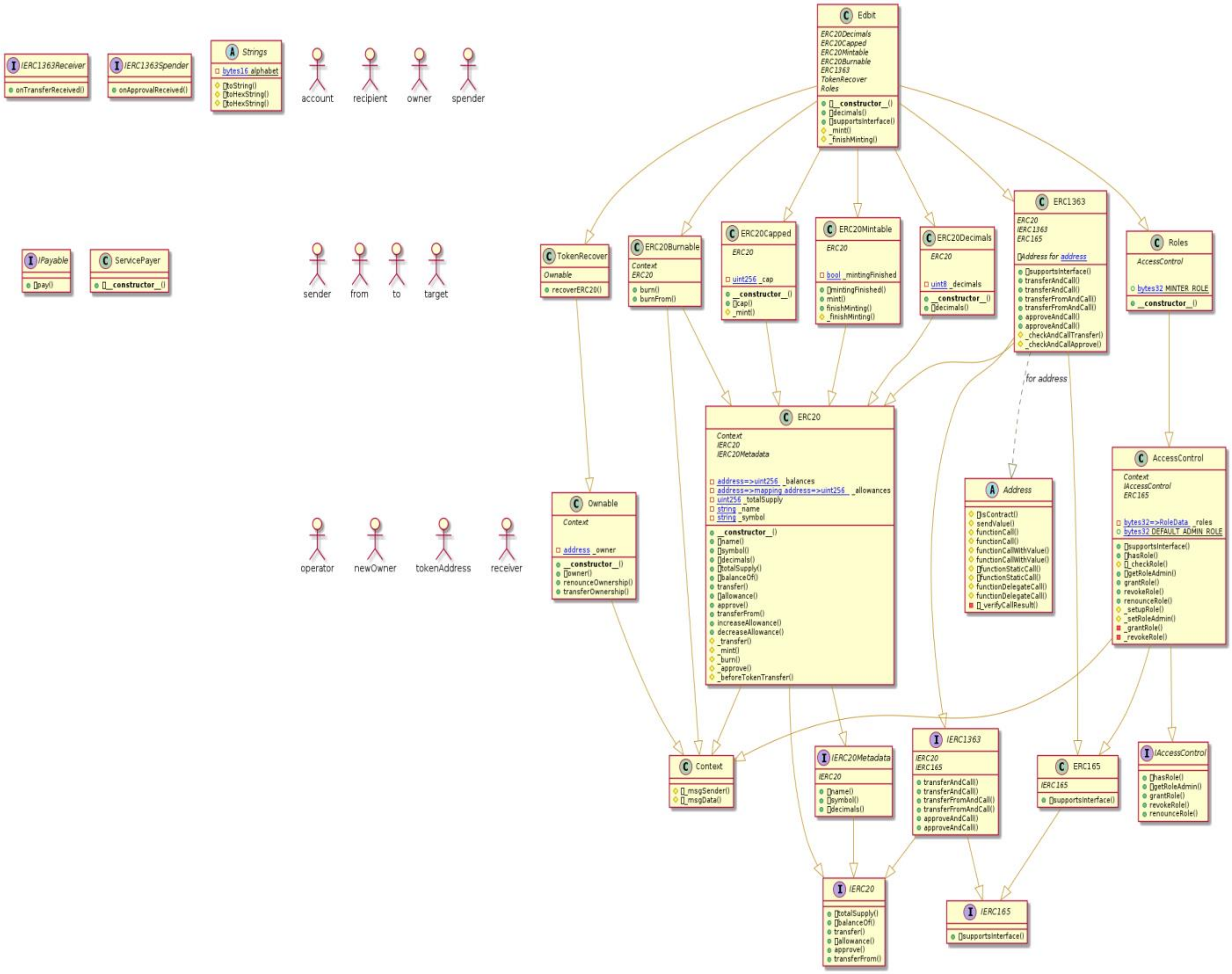
✓ Check sender and value

Result for tests/Edbit Smart Contract2_test.sol
Passed: 5
Failed: 0
Time Taken: 0.48s

5- Call graph


Legend
Internal Call
External Call
Defined Contract
Undefined Contract
ServicePayer
<Constructor>
IPayable (iface)
pay
IAccessControl (iface)
hasRole
getRoleAdmin
grantRole
revokeRole
renounceRole
ERC20Decimals
<Constructor>
decimals
Ownable
<Constructor>
_msgSender
onlyOwner
owner
renounceOwnership
transferOwnership
TokenRecover
recoverERC20
IERC1363Spender (iface)
onApprovalReceived
IERC1363Receiver (iface)
onTransferReceived
IERC1363 (iface)
transferAndCall
transferFromAndCall
approveAndCall
ERC165
supportsInterface
type
IERC165 (iface)
supportsInterface
Address (lib)
sendValue
functionCallWithValue
functionCall
_verifyCallResult
functionStaticCall
isContract
functionDelegateCall
target
staticcall
delegatecall
ERC20Mintable
canMint
mintingFinished
finishMinting
_finishMinting
mint
ERC20
<Constructor>
name
symbol
decimals
balanceOf
transfer
allowance
_mint
totalSupply
transferFrom
_transfer
_beforeTokenTransfer
approve
_approve
decreaseAllowance
_msgSender
increaseAllowance
_burn
Strings (lib)
toString
toHexString
AccessControl
_setupRole
supportsInterface
_checkRole
onlyRole
grantRole
_grantRole
_setRoleAdmin
getRoleAdmin
hasRole
revokeRole
_revokeRole
renounceRole
_msgSender
ERC1363
supportsInterface
transferAndCall
_checkAndCallTransfer
transferFromAndCall
approveAndCall
_checkAndCallApprove
Edbit
<Constructor>
decimals
supportsInterface
_mint
_finishMinting
ERC20Capped
<Constructor>
cap
_mint
Roles
_finishMinting
<Constructor>
_mint
onlyMinter
decimals
Context
_msgSender
_msgData
recipient
isContract
spender
isContract
IERC20Metadata (iface)
name
symbol
decimals
ERC20Burnable
burnFrom
allowance
burn
IERC20 (iface)
totalSupply
balanceOf
transfer
allowance
approve
transferFrom

Unified Modeling Language (UML)



Functions signature

```
Sighash   |   Function Signature
========================
16279055  =>  isContract(address)
39509351  =>  increaseAllowance(address,uint256)
18160ddd  =>  totalSupply()
70a08231  =>  balanceOf(address)
a9059cbb  =>  transfer(address,uint256)
dd62ed3e  =>  allowance(address,address)
095ea7b3  =>  approve(address,uint256)
23b872dd  =>  transferFrom(address,address,uint256)
06fdde03  =>  name()
95d89b41  =>  symbol()
313ce567  =>  decimals()
119df25f  =>  _msgSender()
8b49d47e  =>  _msgData()
a457c2d7  =>  decreaseAllowance(address,uint256)
30e0789e  =>  _transfer(address,address,uint256)
4e6ec247  =>  _mint(address,uint256)
6161eb18  =>  _burn(address,uint256)
104e81ff  =>  _approve(address,address,uint256)
cad3be83  =>  _beforeTokenTransfer(address,address,uint256)
42966c68  =>  burn(uint256)
79cc6790  =>  burnFrom(address,uint256)
355274ea  =>  cap()
24a084df  =>  sendValue(address,uint256)
a0b5ffb0  =>  functionCall(address,bytes)
241b5886  =>  functionCall(address,bytes,string)
2a011594  =>  functionCallWithValue(address,bytes,uint256)
d525ab8a  =>  functionCallWithValue(address,bytes,uint256,string)
c21d36f3  =>  functionStaticCall(address,bytes)
dbc40fb9  =>  functionStaticCall(address,bytes,string)
ee33b7e2  =>  functionDelegateCall(address,bytes)
57387df0  =>  functionDelegateCall(address,bytes,string)
18c2c6a2  =>  _verifyCallResult(bool,bytes,string)
01ffc9a7  =>  supportsInterface(bytes4)
1296ee62  =>  transferAndCall(address,uint256)
4000aea0  =>  transferAndCall(address,uint256,bytes)
d8fbe994  =>  transferFromAndCall(address,address,uint256)
c1d34b89  =>  transferFromAndCall(address,address,uint256,bytes)
3177029f  =>  approveAndCall(address,uint256)
cae9ca51  =>  approveAndCall(address,uint256,bytes)
88a7ca5c  =>  onTransferReceived(address,address,uint256,bytes)
7b04a2d0  =>  onApprovalReceived(address,uint256,bytes)
91d80948  =>  _checkAndCallTransfer(address,address,uint256,bytes)
bf65dd32  =>  _checkAndCallApprove(address,uint256,bytes)
8da5cb5b  =>  owner()
715018a6  =>  renounceOwnership()
f2fde38b  =>  transferOwnership(address)
8980f11f  =>  recoverERC20(address,uint256)
05d2035b  =>  mintingFinished()
40c10f19  =>  mint(address,uint256)
7d64bcb4  =>  finishMinting()
8b16f900  =>  _finishMinting()
```

```
6900a3ae  =>  toString(uint256)
8fba8d5c  =>  toHexString(uint256)
63e1cbea  =>  toHexString(uint256,uint256)
91d14854  =>  hasRole(bytes32,address)
248a9ca3  =>  getRoleAdmin(bytes32)
2f2ff15d  =>  grantRole(bytes32,address)
d547741f  =>  revokeRole(bytes32,address)
36568abe  =>  renounceRole(bytes32,address)
c2985578  =>  foo()
5b7b2c38  =>  _checkRole(bytes32,address)
4fa943a6  =>  _setupRole(bytes32,address)
7612997d  =>  _setRoleAdmin(bytes32,bytes32)
ce2cc1d0  =>  _grantRole(bytes32,address)
2c95bd23  =>  _revokeRole(bytes32,address)
2b66d72e  =>  pay(string)
```

Automatic general report

```
Files Description Table

|  File Name  |  SHA-1 Hash  |
|-------------|--------------|
| /Users/macbook/Desktop/smart contracts/Edbit smart contract.sol |
1e5746a5bcbd81c55dc2cdbab209b7dfdfd92d9b |

 Contracts Description Table

|  Contract  |         Type        |       Bases      |                  |
|
|:----------:|:-------------------:|:----------------:|:----------------:|:--------
-------:|
|     └      |  **Function Name**  |  **Visibility**  |  **Mutability**  |
**Modifiers**  |
||||||
| **IERC20** | Interface |  |||
| └ | totalSupply | External ❗️ |   |NO❗️ |
| └ | balanceOf | External ❗️ |   |NO❗️ |
| └ | transfer | External ❗️ | 🛑  |NO❗️ |
| └ | allowance | External ❗️ |   |NO❗️ |
| └ | approve | External ❗️ | 🛑  |NO❗️ |
| └ | transferFrom | External ❗️ | 🛑  |NO❗️ |
||||||
| **IERC20Metadata** | Interface | IERC20 |||
| └ | name | External ❗️ |   |NO❗️ |
| └ | symbol | External ❗️ |   |NO❗️ |
| └ | decimals | External ❗️ |   |NO❗️ |
||||||
| **Context** | Implementation |  |||
| └ | _msgSender | Internal 🔒 |   | |
| └ | _msgData | Internal 🔒 |   | |
||||||
| **ERC20** | Implementation | Context, IERC20, IERC20Metadata |||
| └ | <Constructor> | Public ❗️ | 🛑  |NO❗️ |
| └ | name | Public ❗️ |   |NO❗️ |
| └ | symbol | Public ❗️ |   |NO❗️ |
| └ | decimals | Public ❗️ |   |NO❗️ |
| └ | totalSupply | Public ❗️ |   |NO❗️ |
| └ | balanceOf | Public ❗️ |   |NO❗️ |
| └ | transfer | Public ❗️ | 🛑  |NO❗️ |
| └ | allowance | Public ❗️ |   |NO❗️ |
| └ | approve | Public ❗️ | 🛑  |NO❗️ |
| └ | transferFrom | Public ❗️ | 🛑  |NO❗️ |
| └ | increaseAllowance | Public ❗️ | 🛑  |NO❗️ |
| └ | decreaseAllowance | Public ❗️ | 🛑  |NO❗️ |
| └ | _transfer | Internal 🔒 | 🛑  | |
| └ | _mint | Internal 🔒 | 🛑  | |
| └ | _burn | Internal 🔒 | 🛑  | |
| └ | _approve | Internal 🔒 | 🛑  | |
| └ | _beforeTokenTransfer | Internal 🔒 | 🛑  | |
||||||
| **ERC20Burnable** | Implementation | Context, ERC20 |||
| └ | burn | Public ❗️ | 🛑  |NO❗️ |
| └ | burnFrom | Public ❗️ | 🛑  |NO❗️ |
||||||
| **ERC20Capped** | Implementation | ERC20 |||
| └ | <Constructor> | Public ❗️ | 🛑  |NO❗️ |
| └ | cap | Public ❗️ |   |NO❗️ |
| └ | _mint | Internal 🔒 | 🛑  | |
||||||
| **Address** | Library |  |||
| └ | isContract | Internal 🔒 |   | |
| └ | sendValue | Internal 🔒 | 🛑  | |
| └ | functionCall | Internal 🔒 | 🛑  | |
| └ | functionCall | Internal 🔒 | 🛑  | |
| └ | functionCallWithValue | Internal 🔒 | 🛑  | |
| └ | functionCallWithValue | Internal 🔒 | 🛑  | |
| └ | functionStaticCall | Internal 🔒 |   | |
| └ | functionStaticCall | Internal 🔒 |   | |
| └ | functionDelegateCall | Internal 🔒 | 🛑  | |
| └ | functionDelegateCall | Internal 🔒 | 🛑  | |
| └ | _verifyCallResult | Private 🔐 |   | |
||||||
| **IERC165** | Interface |  |||
| └ | supportsInterface | External ❗️ |   |NO❗️ |
||||||
| **ERC165** | Implementation | IERC165 |||
| └ | supportsInterface | Public ❗️ |   |NO❗️ |
||||||
| **IERC1363** | Interface | IERC20, IERC165 |||
| └ | transferAndCall | External ❗️ | 🛑  |NO❗️ |
| └ | transferAndCall | External ❗️ | 🛑  |NO❗️ |
| └ | transferFromAndCall | External ❗️ | 🛑  |NO❗️ |
| └ | transferFromAndCall | External ❗️ | 🛑  |NO❗️ |
| └ | approveAndCall | External ❗️ | 🛑  |NO❗️ |
| └ | approveAndCall | External ❗️ | 🛑  |NO❗️ |
||||||
| **IERC1363Receiver** | Interface |  |||
| └ | onTransferReceived | External ❗️ | 🛑  |NO❗️ |
||||||
| **IERC1363Spender** | Interface |  |||
| └ | onApprovalReceived | External ❗️ | 🛑  |NO❗️ |
||||||
| **ERC1363** | Implementation | ERC20, IERC1363, ERC165 |||
| └ | supportsInterface | Public ❗️ |   |NO❗️ |
| └ | transferAndCall | Public ❗️ | 🛑  |NO❗️ |
| └ | transferAndCall | Public ❗️ | 🛑  |NO❗️ |
| └ | transferFromAndCall | Public ❗️ | 🛑  |NO❗️ |
| └ | transferFromAndCall | Public ❗️ | 🛑  |NO❗️ |
| └ | approveAndCall | Public ❗️ | 🛑  |NO❗️ |
| └ | approveAndCall | Public ❗️ | 🛑  |NO❗️ |
| └ | _checkAndCallTransfer | Internal 🔒 | 🛑  | |
| └ | _checkAndCallApprove | Internal 🔒 | 🛑  | |
||||||
| **Ownable** | Implementation | Context |||
| └ | <Constructor> | Public ❗️ | 🛑  |NO❗️ |
| └ | owner | Public ❗️ |   |NO❗️ |
| └ | renounceOwnership | Public ❗️ | 🛑  | onlyOwner |
| └ | transferOwnership | Public ❗️ | 🛑  | onlyOwner |
||||||
| **TokenRecover** | Implementation | Ownable |||
| └ | recoverERC20 | Public ❗️ | 🛑  | onlyOwner |
||||||
| **ERC20Decimals** | Implementation | ERC20 |||
| └ | <Constructor> | Public ❗️ | 🛑  |NO❗️ |
| └ | decimals | Public ❗️ |   |NO❗️ |
||||||
| **ERC20Mintable** | Implementation | ERC20 |||
| └ | mintingFinished | External ❗️ |   |NO❗️ |
| └ | mint | External ❗️ | 🛑  | canMint |
| └ | finishMinting | External ❗️ | 🛑  | canMint |
| └ | _finishMinting | Internal 🔒 | 🛑  | |
||||||
| **Strings** | Library |  |||
| └ | toString | Internal 🔒 |   | |
| └ | toHexString | Internal 🔒 |   | |
| └ | toHexString | Internal 🔒 |   | |
||||||
| **IAccessControl** | Interface |  |||
| └ | hasRole | External ❗️ |   |NO❗️ |
| └ | getRoleAdmin | External ❗️ |   |NO❗️ |
| └ | grantRole | External ❗️ | 🛑  |NO❗️ |
| └ | revokeRole | External ❗️ | 🛑  |NO❗️ |
| └ | renounceRole | External ❗️ | 🛑  |NO❗️ |
||||||
| **AccessControl** | Implementation | Context, IAccessControl, ERC165 |||
| └ | supportsInterface | Public ❗️ |   |NO❗️ |
| └ | hasRole | Public ❗️ |   |NO❗️ |
| └ | _checkRole | Internal 🔒 |   | |
| └ | getRoleAdmin | Public ❗️ |   |NO❗️ |
| └ | grantRole | Public ❗️ | 🛑  | onlyRole |
| └ | revokeRole | Public ❗️ | 🛑  | onlyRole |
| └ | renounceRole | Public ❗️ | 🛑  |NO❗️ |
| └ | _setupRole | Internal 🔒 | 🛑  | |
| └ | _setRoleAdmin | Internal 🔒 | 🛑  | |
| └ | _grantRole | Private 🔐 | 🛑  | |
| └ | _revokeRole | Private 🔐 | 🛑  | |
||||||
| **Roles** | Implementation | AccessControl |||
| └ | <Constructor> | Public ❗️ | 🛑  |NO❗️ |
||||||
| **IPayable** | Interface |  |||
| └ | pay | External ❗️ |  💵 |NO❗️ |
||||||
| **ServicePayer** | Implementation |  |||
| └ | <Constructor> | Public ❗️ |  💵 |NO❗️ |
||||||
| **Edbit** | Implementation | ERC20Decimals, ERC20Capped, ERC20Mintable,
ERC20Burnable, ERC1363, TokenRecover, Roles |||
| └ | <Constructor> | Public ❗️ |  💵 | ERC20 ERC20Decimals ERC20Capped |
| └ | decimals | Public ❗️ |   |NO❗️ |
| └ | supportsInterface | Public ❗️ |   |NO❗️ |
| └ | _mint | Internal 🔒 | 🛑  | onlyMinter |
| └ | _finishMinting | Internal 🔒 | 🛑  | 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 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.