Smart Contract

Security Audit V2

Chronicum ERC20 Smart Contract

16/3/2022

OnlyUp Capital

Table of Contents

Table of Contents

Background

Project Information

- Smart Contract Information
- Executive Summary

File and Function Level Report

- **File in Scope:**

Issues Checking Status

- Severity Definitions
- Audit Findings

Automatic testing

- Testing proves
- Inheritance graph
- Call graph

Unified Modeling Language (UML)

Functions signature

Automatic general report

Conclusion

Disclaimer

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**: Avalanche C-Chain
- **Contract Address**: 0x04425384B244bE1D571e2c559afC15ecEc1dEDcB
- **Code:**

https://testnet.snowtrace.io/address/0x04425384b244be1d571e2c559afc15ecec1dedcb#code

Smart Contract Information

- Name: CHRO
- Max Supply: 300,000,000
- Holders:
- Total transactions:

Contracts address deployed to test net (AVAX)

Chronicum Smart contract on AVAX test net to test write functions by the auditor.

https://testnet.snowtrace.io/address/0x04425384b244be1d571e2c559afc15ecec1dedcb

Executive Summary

According to our assessment, the customer`s solidity smart contract is **Well-Secured**. The team acknowledged and fixed any severe 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 1 critical, 0 high, 0 medium, 3 low, 0 very low-level issues and 1 note in all solidity files of the contract

The files:

CHRO.sol

File and Function Level Report

File in Scope:

Contract Name	SHA 256 hash	Contract Address
CHRO.sol	**a415ed1b7082f419bafcd58a6dd3f7afe3e4a2ed44f4d6be1c3955aea79d8414**	0x04425384B244bE1D571e2c559afC15ecEc1dEDcB

- Contract: CHRO
- Inherit: ERC20, ERC20Permit, ERC20Votes, ERC20Pausable, 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**
checkpoints	✓	Read / public	**Passed**
delegates	✓	Read / public	**Passed**
decimal	✓	Read / public	**Passed**
balanceOf	✓	Read / public	**Passed**
Owner	✓	Read / public	**Passed**
DOMAIN_SEPARATOR	✓	Read / public	**Passed**
getPastTotalSupply	✓	Read / public	**Passed**
getPastVotes	✓	Read / public	**Passed**
getVotes	✓	Read / public	**Passed**
nonces	✓	Read / public	**Passed**

numCheckpoints	✓	Read / public	**Passed**
paused	✓	Read / public	**Passed**
totalSupply	✓	Read / public	**Passed**
allowance	✓	Read / public	**Passed**
mint	✓	Write / public	**Passed**
approve	✓	Write / public	**Passed**
transfer	✓	Write / public	**Passed**
burn	✓	Write / public	**Passed**
increaseAllowance	✓	Write / public	**Passed**
decreaseAllowance	✓	Write / public	**Passed**
permit	✓	Write / public	**Passed**
transferOwnership	✓	Write / public	**Passed**
transferFrom	✓	Write / public	**Passed**
unPause	✓	Write / public	**Passed**
pause	✓	Write / public	**Passed**
delegateBySig	✓	Write / public	**Passed**
renounceOwnership	✓	Write / public	**Passed**
delegate	✓	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**
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:

#The owner can burn all total supply

Description

The owner has the ability to burn all total supply including holders tokens without their permission. This represents a risk for the users because in the case that the function is called, their funds will be burned without his/her permission.

```
function burn(
    address _account,
    uint256 _amount
  ) public onlyOwner {
    _burn(_account, _amount);
  }
```

Remediation

There are two options to remediate the risk. Make this function internal so no one can control it . The second one is making the burn function burn from the total supply not from user's funds which is recommended by the auditor.

Status: Closed. Fixed in version2.

High:

No High severity vulnerabilities were found

Medium:

No Medium severity vulnerabilities were found

Low:

#Pragam version not fixed

Description

It is a good practice to lock the solidity version for a live deployment (use 0.8.6 instead of ^0.8.0). contracts should be deployed with the same compiler version and flags that they have been tested the most with. Locking the pragma helps ensure that contracts do not accidentally get deployed using, for example, the latest compiler which may have higher risks of undiscovered bugs. Contracts may also be deployed by others and the pragma indicates the compiler version intended by the original authors.

Remediation

Remove the ^ sign to lock the pragma version.

Status: Closed. Fixed in version2.

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

Description

The owner can pause and un pause the contract.

```
function pause() public onlyOwner {
        _pause();
    }

    function unpause() public onlyOwner {
        _unpause();
    }
```

Remediation

Make these functions internal in next version or the team should announce the investors before pause and un pause to give them time if they want to do anything.
P.S: This issue is common to the majority of pausable smart contracts.

Status: Acknowledged.

#Missing zero address validation

Description
When the owner wants to mint for the investors it has to check for the zero address to make owner didn't mint to the burn address. Otherwise, the mint function will act like the burn function.

```
function mint(
    address _to,
    uint256 _amount
  ) public onlyOwner {
    require(totalSupply() + _amount <= MAX_SUPPLY, "Exceeded maximum supply");
    _mint(_to, _amount);
  }
```

Remediation
Use the require statement to check for zero addresses.

```
require(_to != address(0), "Not Mint for the zero address");
```

Status: Closed. Fixed in version2.

Very Low:

No Very Low severity vulnerabilities were found.

Notes:

#Unnecessary import of ERC20, and ERC20Permit libraries

Description
The main contract inherits: ERC20, ERC20Permit, ERC20Votes, ERC20Pausable, Ownable, and ERC20Votes is already import ERC20, ERC20Permit libraries so no need to import it again in the main contract.

Remediation
Remove address library for the main contract save some gas fees.

Status: Acknowledged

Automatic Testing

1- Check for security

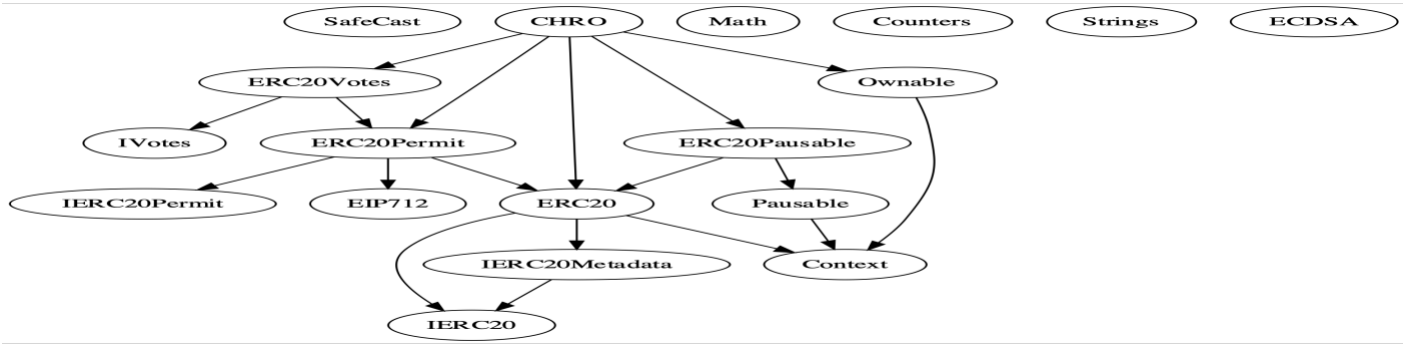
a415ed1b7082f419bafcd58a6dd3f7afe3e4a2ed44f4d6be1c3955aea79d8414

File: CHRO.sol | Language: solidity | Size: 2328 bytes | Date: 2022-03-16T08:13:54.913Z

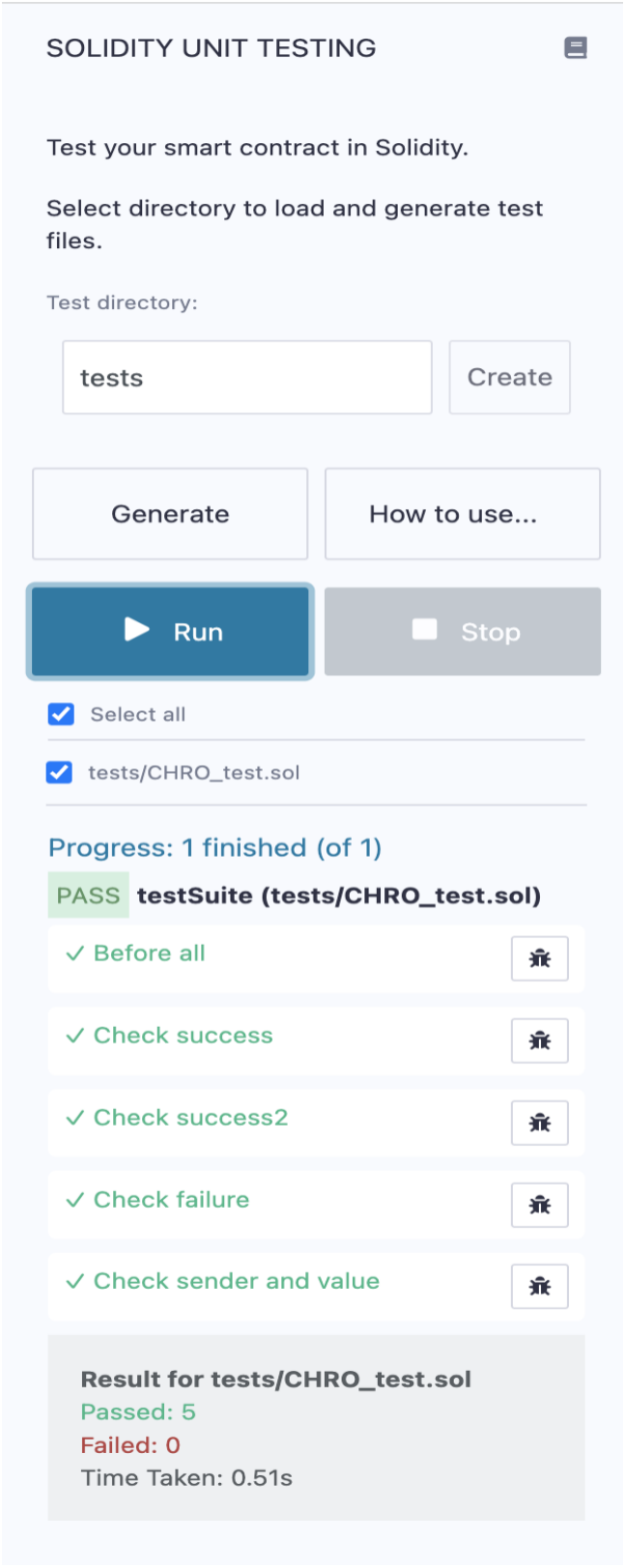
Critical	High	Medium	Low	Note
0	0	0	0	0

2- SOLIDITY STATIC ANALYSIS

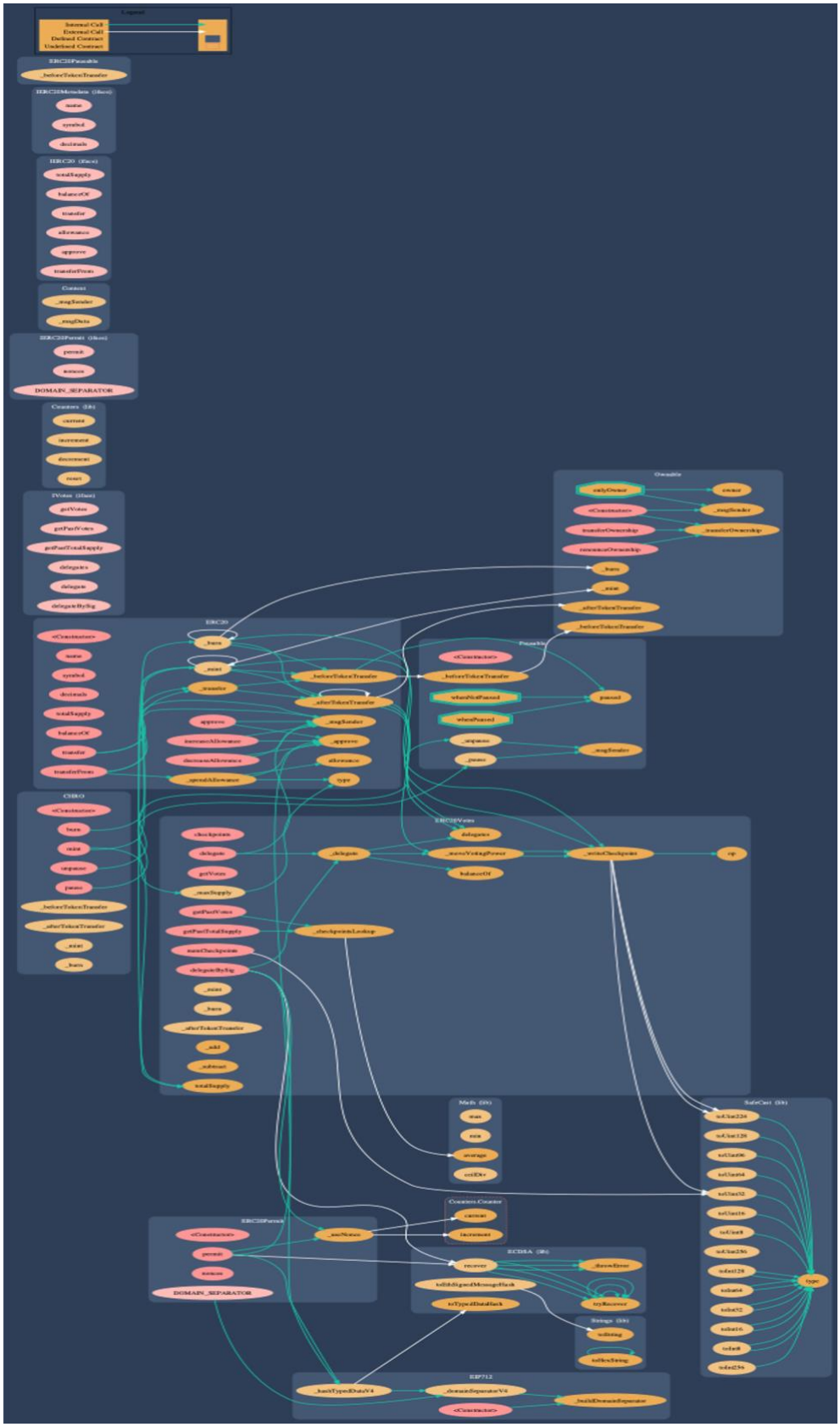
3- Inheritance graph



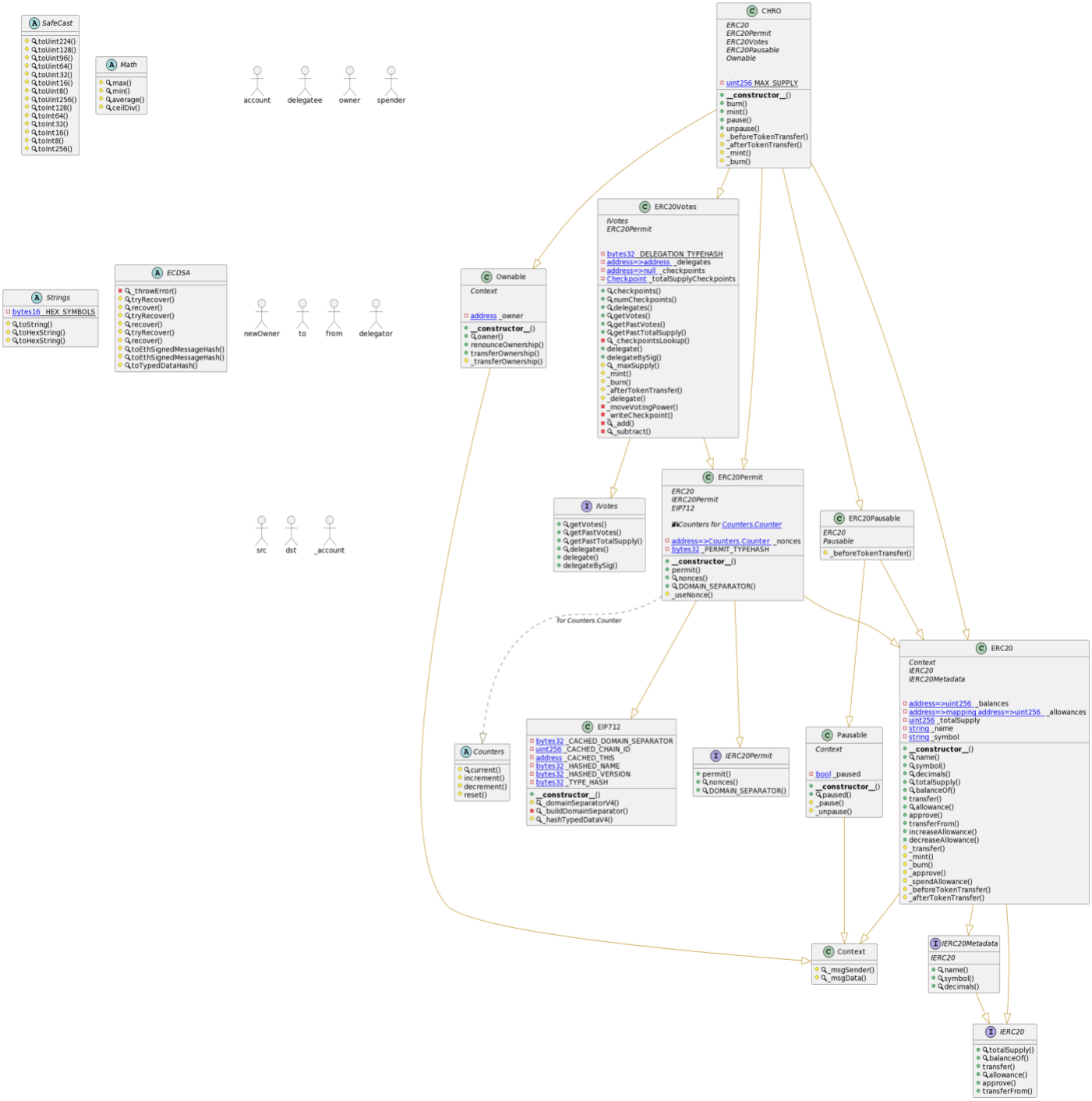
4- SOLIDITY UNIT TESTING



5- Call graph



Unified Modeling Language (UML)



Functions signature

```
Sighash   |   Function Signature
========================
39509351  =>  increaseAllowance(address,uint256)
5bb79860  =>  toUint224(uint256)
809fdd33  =>  toUint128(uint256)
1cf887fc  =>  toUint96(uint256)
2665fad0  =>  toUint64(uint256)
c8193255  =>  toUint32(uint256)
9374068f  =>  toUint16(uint256)
0cc4681e  =>  toUint8(uint256)
fdcf791b  =>  toUint256(int256)
dd2a0316  =>  toInt128(int256)
d6bd32aa  =>  toInt64(int256)
9c6f59be  =>  toInt32(int256)
cf65b4d3  =>  toInt16(int256)
f136dc02  =>  toInt8(int256)
dfbe873b  =>  toInt256(uint256)
9ab24eb0  =>  getVotes(address)
3a46b1a8  =>  getPastVotes(address,uint256)
8e539e8c  =>  getPastTotalSupply(uint256)
587cde1e  =>  delegates(address)
5c19a95c  =>  delegate(address)
c3cda520  =>  delegateBySig(address,uint256,uint256,uint8,bytes32,bytes32)
6d5433e6  =>  max(uint256,uint256)
7ae2b5c7  =>  min(uint256,uint256)
2b7423ab  =>  average(uint256,uint256)
9cb35327  =>  ceilDiv(uint256,uint256)
ad04a8d1  =>  current(Counter)
e2bee435  =>  increment(Counter)
854ec98e  =>  decrement(Counter)
440d212a  =>  reset(Counter)
6900a3ae  =>  toString(uint256)
8fba8d5c  =>  toHexString(uint256)
63e1cbea  =>  toHexString(uint256,uint256)
5e2ffa14  =>  _throwError(RecoverError)
c6edd8a7  =>  tryRecover(bytes32,bytes)
19045a25  =>  recover(bytes32,bytes)
628f98cc  =>  tryRecover(bytes32,bytes32,bytes32)
bf2fe7fd  =>  recover(bytes32,bytes32,bytes32)
4d78da76  =>  tryRecover(bytes32,uint8,bytes32,bytes32)
c2bf17b0  =>  recover(bytes32,uint8,bytes32,bytes32)
918a15cf  =>  toEthSignedMessageHash(bytes32)
92bd87b5  =>  toEthSignedMessageHash(bytes)
7df7a71c  =>  toTypedDataHash(bytes32,bytes32)
7b134b4c  =>  _domainSeparatorV4()
112794f2  =>  _buildDomainSeparator(bytes32,bytes32,bytes32)
c8f1ecd8  =>  _hashTypedDataV4(bytes32)
d505accf  =>  permit(address,address,uint256,uint256,uint8,bytes32,bytes32)
7ecebe00  =>  nonces(address)
3644e515  =>  DOMAIN_SEPARATOR()
119df25f  =>  _msgSender()
8b49d47e  =>  _msgData()
8da5cb5b  =>  owner()
```

```
715018a6  =>  renounceOwnership()
f2fde38b  =>  transferOwnership(address)
d29d44ee  =>  _transferOwnership(address)
5c975abb  =>  paused()
320b2ad9  =>  _pause()
fc8234cb  =>  _unpause()
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()
a457c2d7  =>  decreaseAllowance(address,uint256)
30e0789e  =>  _transfer(address,address,uint256)
4e6ec247  =>  _mint(address,uint256)
6161eb18  =>  _burn(address,uint256)
104e81ff  =>  _approve(address,address,uint256)
1532335e  =>  _spendAllowance(address,address,uint256)
cad3be83  =>  _beforeTokenTransfer(address,address,uint256)
8f811a1c  =>  _afterTokenTransfer(address,address,uint256)
35d11de3  =>  _useNonce(address)
f1127ed8  =>  checkpoints(address,uint32)
6fcfff45  =>  numCheckpoints(address)
db263f39  =>  _checkpointsLookup(Checkpoint256[],uint256)
22f4596f  =>  _maxSupply()
a28a42b3  =>  _delegate(address,address)
82851b84  =>  _moveVotingPower(address,address,uint256)
5c3188b4  =>  _writeCheckpoint(Checkpoint256[],function(uint256,uint256)
3d0316c3  =>  _add(uint256,uint256)
880bf496  =>  _subtract(uint256,uint256)
9dc29fac  =>  burn(address,uint256)
40c10f19  =>  mint(address,uint256)
8456cb59  =>  pause()
3f4ba83a  =>  unpause()
```

Automatic general report

```
Files Description Table

|  File Name  |  SHA-1 Hash  |
|-------------|--------------|
| /Users/macbook/Desktop/smart contracts/CHRO.sol |
18b8509424dd7c805b78234ac45a51e8eb3d132f |

 Contracts Description Table

|  Contract  |         Type        |       Bases      |                  |
|
|:----------:|:-------------------:|:----------------:|:----------------:|:--------
-------:|
|     └      |  **Function Name**  |  **Visibility**  |  **Mutability**  |
**Modifiers**  |
||||||
| **SafeCast** | Library |  |||
| └ | toUint224 | Internal 🔒 |   | |
| └ | toUint128 | Internal 🔒 |   | |
| └ | toUint96 | Internal 🔒 |   | |
| └ | toUint64 | Internal 🔒 |   | |
| └ | toUint32 | Internal 🔒 |   | |
| └ | toUint16 | Internal 🔒 |   | |
| └ | toUint8 | Internal 🔒 |   | |
| └ | toUint256 | Internal 🔒 |   | |
| └ | toInt128 | Internal 🔒 |   | |
| └ | toInt64 | Internal 🔒 |   | |
| └ | toInt32 | Internal 🔒 |   | |
| └ | toInt16 | Internal 🔒 |   | |
| └ | toInt8 | Internal 🔒 |   | |
| └ | toInt256 | Internal 🔒 |   | |
||||||
| **IVotes** | Interface |  |||
| └ | getVotes | External ❗ |   |NO❗ |
| └ | getPastVotes | External ❗ |   |NO❗ |
| └ | getPastTotalSupply | External ❗ |   |NO❗ |
| └ | delegates | External ❗ |   |NO❗ |
| └ | delegate | External ❗ | 🛑 |NO❗ |
| └ | delegateBySig | External ❗ | 🛑 |NO❗ |
||||||
| **Math** | Library |  |||
| └ | max | Internal 🔒 |   | |
| └ | min | Internal 🔒 |   | |
| └ | average | Internal 🔒 |   | |
| └ | ceilDiv | Internal 🔒 |   | |
||||||
| **Counters** | Library |  |||
| └ | current | Internal 🔒 |   | |
| └ | increment | Internal 🔒 | 🛑 | |
| └ | decrement | Internal 🔒 | 🛑 | |
| └ | reset | Internal 🔒 | 🛑 | |
||||||
```

Strings	Library			
└	toString	Internal 🔒		
└	toHexString	Internal 🔒		
└	toHexString	Internal 🔒		
ECDSA	Library			
└	_throwError	Private 🔐		
└	tryRecover	Internal 🔒		
└	recover	Internal 🔒		
└	tryRecover	Internal 🔒		
└	recover	Internal 🔒		
└	tryRecover	Internal 🔒		
└	recover	Internal 🔒		
└	toEthSignedMessageHash	Internal 🔒		
└	toEthSignedMessageHash	Internal 🔒		
└	toTypedDataHash	Internal 🔒		
EIP712	Implementation			
└	<Constructor>	Public ❗	🛑	NO❗
└	_domainSeparatorV4	Internal 🔒		
└	_buildDomainSeparator	Private 🔐		
└	_hashTypedDataV4	Internal 🔒		
IERC20Permit	Interface			
└	permit	External ❗	🛑	NO❗
└	nonces	External ❗		NO❗
└	DOMAIN_SEPARATOR	External ❗		NO❗
Context	Implementation			
└	_msgSender	Internal 🔒		
└	_msgData	Internal 🔒		
Ownable	Implementation	Context		
└	<Constructor>	Public ❗	🛑	NO❗
└	owner	Public ❗		NO❗
└	renounceOwnership	Public ❗	🛑	onlyOwner
└	transferOwnership	Public ❗	🛑	onlyOwner
└	_transferOwnership	Internal 🔒	🛑	
Pausable	Implementation	Context		
└	<Constructor>	Public ❗	🛑	NO❗
└	paused	Public ❗		NO❗
└	_pause	Internal 🔒	🛑	whenNotPaused
└	_unpause	Internal 🔒	🛑	whenPaused
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❗
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 🔒	🛑	
└	_spendAllowance	Internal 🔒	🛑	
└	_beforeTokenTransfer	Internal 🔒	🛑	
└	_afterTokenTransfer	Internal 🔒	🛑	
ERC20Permit	Implementation	ERC20, IERC20Permit, EIP712		
└	<Constructor>	Public ❗	🛑	EIP712
└	permit	Public ❗	🛑	NO❗
└	nonces	Public ❗		NO❗
└	DOMAIN_SEPARATOR	External ❗		NO❗
└	_useNonce	Internal 🔒	🛑	
ERC20Votes	Implementation	IVotes, ERC20Permit		
└	checkpoints	Public ❗		NO❗
└	numCheckpoints	Public ❗		NO❗
└	delegates	Public ❗		NO❗
└	getVotes	Public ❗		NO❗
└	getPastVotes	Public ❗		NO❗
└	getPastTotalSupply	Public ❗		NO❗
└	_checkpointsLookup	Private 🔐		
└	delegate	Public ❗	🛑	NO❗
└	delegateBySig	Public ❗	🛑	NO❗
└	_maxSupply	Internal 🔒		
└	_mint	Internal 🔒	🛑	
└	_burn	Internal 🔒	🛑	
└	_afterTokenTransfer	Internal 🔒	🛑	
└	_delegate	Internal 🔒	🛑	
└	_moveVotingPower	Private 🔐	🛑	
└	_writeCheckpoint	Private 🔐	🛑	
└	_add	Private 🔐		
└	_subtract	Private 🔐		
ERC20Pausable	Implementation	ERC20, Pausable		
└	_beforeTokenTransfer	Internal 🔒	🛑	
CHRO	Implementation	ERC20, ERC20Permit, ERC20Votes, ERC20Pausable,

Ownable |||
└	<Constructor>	Public ❗	🛑	ERC20 ERC20Permit
└	burn	Public ❗	🛑	onlyOwner
└	mint	Public ❗	🛑	onlyOwner
└	pause	Public ❗	🛑	onlyOwner
└	unpause	Public ❗	🛑	onlyOwner
└	_beforeTokenTransfer	Internal 🔒	🛑	
└	_afterTokenTransfer	Internal 🔒	🛑	
└	_mint	Internal 🔒	🛑	
└	_burn	Internal 🔒	🛑	

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

Conclusion

The contracts are written systematically. No critical issues were found, as such, the contract is cleared 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. The team will go into more detail on this in the 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 OnlyUp Capital and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) (OnlyUp Capital) owe no duty of care towards you or any other person, nor does OnlyUp Capital 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 OnlyUp Capital 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, OnlyUp Capital hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against OnlyUp Capital, 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.