AudiTBlock

MewCoin

AuditReport

Low-Risk	Medium-Risk	High-Risk
Low-risk code	Medium-risk code	High-risk code

Types of Severities

High

A high-severity issue or vulnerability means that your smart contract can be exploited. Issues on this level are critical to the smart contract's performance or functionality, and we recommend these issues be fixed before moving to a live environment.

Medium

The issues marked as medium severity usually arise because of errors and deficiencies in the smart contract code. Issues on this level could potentially bring problems, and they should still be fixed.

Low

Low-level severity issues can cause minor impact and or are just warnings that can remain unfixed for now. It would be better to fix these issues at some point in the future.

Informational

These are severity issues that indicate an improvement request, a general question, a cosmetic or documentation error, or a request for information. There is low-to-no impact.

Techniques and Methods

The overall quality of code.

- Use of best practices.
- Code documentation and comments match logic and expected behavior.
- Token distribution and calculations are as per the intended behavior mentioned in the whitepaper.
- implementation of ERC-20 token standards.
- Efficient use of gas.
- Code is safe from re-entrance and other vulnerabilities.

The following techniques, methods, and tools were used to review all the smart contracts.

Structural Analysis

In this step, we have analyzed the design patterns and structure of smart contracts. A thorough check was done to ensure the smart contract is structured in a way that will not result in future problems.

Static Analysis

Static analysis of smart contracts was done to identify contract vulnerabilities. In this step, a series of automated tools are used to test the security of smart contracts.

Code Review / Manual Analysis

Manual analysis or review of code was done to identify new vulnerabilities or verify the vulnerabilities found during the static analysis. Contracts were completely manually analyzed, and their logic was checked and compared with the one described in the whitepaper. Besides, the results of the automated analysis were manually verified.

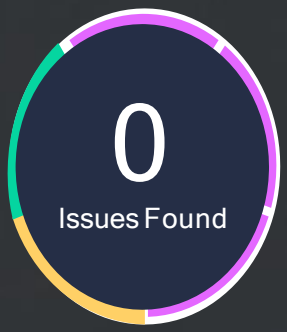
Gas Consumption

In this step, we have checked the behavior of smart contracts in production. Checks were done to know how much gas gets consumed and the possibilities of optimization of code to reduce gas consumption.

Tools and Platforms Used for Audit

Remix IDE, Truffle, Truffle Team, Solhint, Mythril, Slither, Solidity statistic analysis.

Name	MewCoin
Website	mew-coin.com
Method	Manual Review, Functional Testing, Automated Testing etc.
Scope of Audit	The scope of this audit was to analyze the contract codebase for quality, security, and correctness.
Audit Team	AuditBlock



High Medium

Low Informational

	High	Medium	Low	Informational
Open Issues	0	0	0	0
Acknowledged Issues	0	0	0	0
Partially Resolved Issues	0	0	0	0
Resolved Issues	0	0	0	0

ID	File Name	Audit Status
10024	MewCoin.sol	Pass

Smart Contract Weakness Classification (SWC) Vulnerabilities for Attacks

- Re-entrancy
- Timestamp Dependence
- Gas Limit and Loops
- Exception Disorder
- Gasless Send
- Use of tx.origin
- Compiler version not fixed
- Address hardcoded
- Divide before multiply
- Integer overflow/underflow
- Dangerous strict equalities
- Tautology or contradiction
- Missing Zero Address Validation
- Return values of low-level calls
- Revert/require functions
- Private modifier
- Using block.timestamp
- Multiple Sends
- Using SHA3
- Using suicide
- Using throw
- Using inline assembly

Phase 1

High Severity Issues

No issues found

Medium Severity Issues

No issues found

Low Severity Issues

No issues found

Informational Severity Issues

No issues found

Phase 2

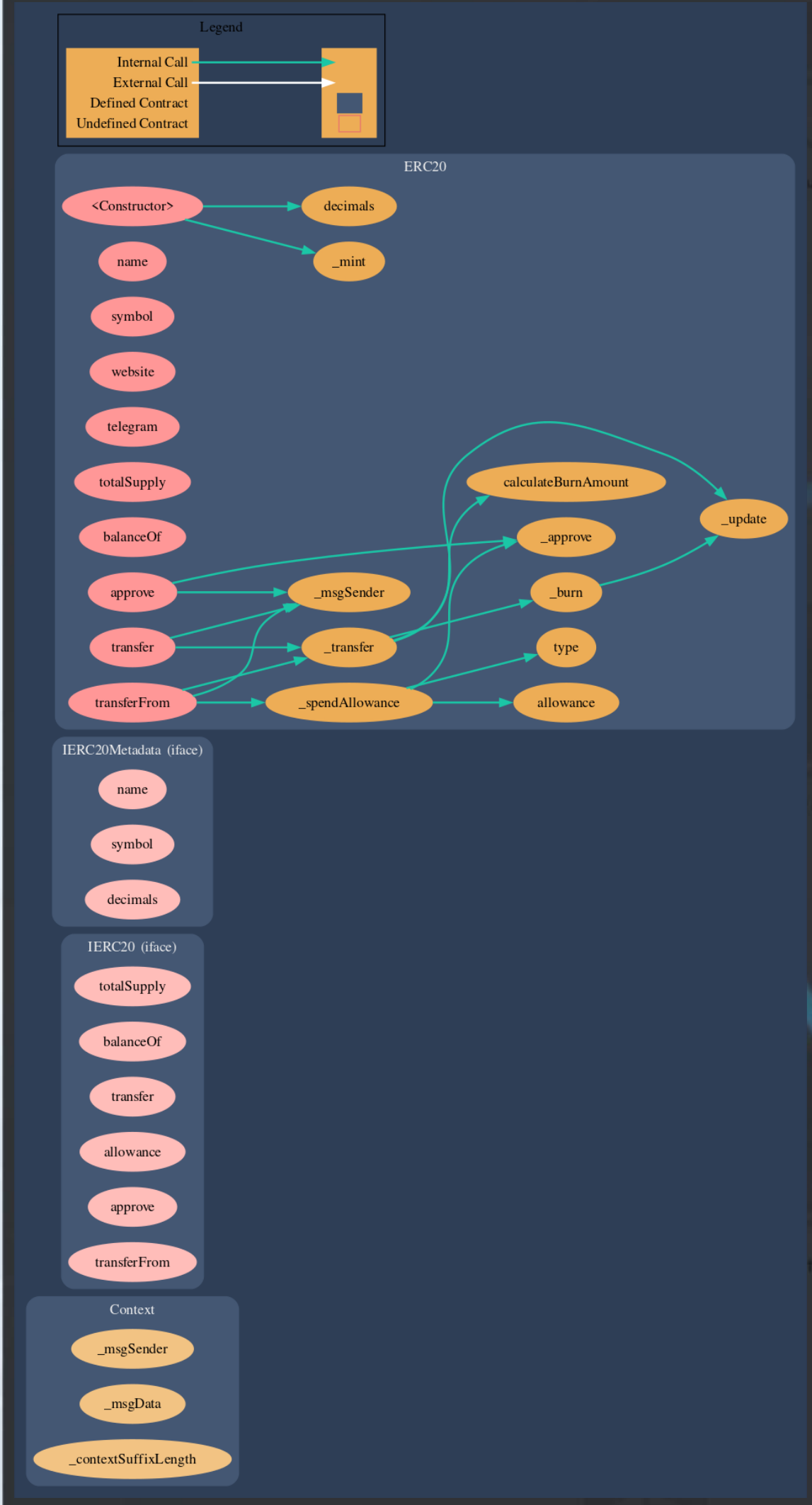
MewCoin.nativeToToken(uint256,uint256) (contracts/MewCoin.sol#706-714) performs a multiplication on the result of a division:
Version constraint ^0.8.20 contains known severe issues (https://solidity.readthedocs.io/en/latest/bugs.html)
- VerbatimInvalidDeduplication
- FullInlinerNonExpressionSplitArgumentEvaluationOrder
- MissingSideEffectsOnSelectorAccess.

It is used by:
- contracts/MewCoin.sol#6
- contracts/MewCoin.sol#171
- contracts/MewCoin.sol#202
- contracts/MewCoin.sol#284
- contracts/MewCoin.sol#310

Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

INFO:Detectors:
ERC20.constructor() (contracts/MewCoin.sol#326-335) uses literals with too many digits:
- totalInitialSupply = 1000000000 * 10 ** uint256(decimals()) (contracts/MewCoin.sol#332)

Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits


Legend
Internal Call
External Call
Defined Contract
Undefined Contract
ERC20
<Constructor>
decimals
_mint
name
symbol
website
telegram
totalSupply
balanceOf
approve
transfer
transferFrom
_msgSender
_transfer
_spendAllowance
calculateBurnAmount
_approve
_burn
type
allowance
_update
IERC20Metadata (iface)
name
symbol
decimals
IERC20 (iface)
totalSupply
balanceOf
transfer
allowance
approve
transferFrom
Context
_msgSender
_msgData
_contextSuffixLength

Phase 3

Closing Summary

In this report, we have considered the security of this **MewCoin** We performed our audit according to the procedure described above.

No issues were identified during the audit and classified by severity. Recommendations and best practices were provided to improve code quality and security posture. The team has acknowledged all findings.

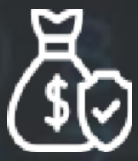
Disclaimer

AuditBlock does not provide security warranties, investment advice, or endorsements of any platform. This audit does not guarantee the security or correctness of the audited smart contracts. The statements made in this document should not be interpreted as investment or legal advice. The authors are not liable for any decisions made based on the information in this document. Securing smart contracts is an ongoing process. A single audit is not sufficient. We recommend that the platform's development team implement a bug bounty program to encourage further analysis of the smart contract by other third parties

AuditBlock

AuditBlock is a blockchain security company that provides professional services and solutions for securing blockchain projects. They specialize in smart contract audits on various blockchains and offer a range of services

100+
Audits Completed



$1M
Secured

100K
Lines of Code Audited

https://auditblock.report/

https://t.me/Auditblock

https://github.com/AuditBlock

https://twitter.com/oAuditBlock