EVM
SMART AUDIT

SMART CONTRACT SECURITY AUDIT

Smart Contract Security Audit

Hybrid static & AI-assisted analysis

MMSCash

Chain: bsc **Compiler:** v0.8.24+commit.e11b9ed9

Address: `0xc3bfbfd5abbf5aab5e336f5f39b079514025405b`



Low Risk

12 detailed findings · 0 severity ≥ high

Informational **4** Gas **8**

EXECUTIVE SUMMARY

Executive Summary

Static analysis of MMSCash produced 12 finding(s): 4 informational, 8 gas. Review the high-severity items first. Enable AI analysis for deeper business-logic and economic-attack coverage.

Critical	0
High	0
Medium	0
Low	0
Informational	4
Gas	8



Scope & Methodology

Contract	**MMSCash**
Chain	bsc
Address	`0xc3bfbfd5abbf5aab5e336f5f39b079514025405b`
Compiler	v0.8.24+commit.e11b9ed9
Files analyzed	1

ANALYSIS ENGINES

- Built-in detectors — Enabled
- Slither — Not run
- AI deep analysis — Not run

Contract Overview

621	3	30
Lines of code	Contracts	Functions

0	6	1
External calls	State variables	Files analyzed

Contracts

CONTRACT	KIND	INHERITS
Context	abstract	—
IERC20	interface	—
IERC20Errors	interface	—
IERC721Errors	interface	—
IERC1155Errors	interface	—
IERC20Metadata	interface	IERC20
ERC20	abstract	Context, IERC20, IERC20Metadata, IERC20Errors
MMSCash	contract	ERC20

FUNCTION SURFACE



External	**9**
Public	**11**
Internal	**10**
Private	**0**

Payable: 0 | View / Pure: 15

Externally exposed: 20

STANDARDS COVERAGE

Standards Coverage

MMSCash was reviewed against the following industry checklists; flagged items map to findings in this report.

OWASP Smart Contract Top 10 (2025)

9/10 clear

✓	SC01	Access Control Vulnerabilities	No issues
✓	SC02	Price Oracle Manipulation	No issues
✓	SC03	Logic Errors	No issues
✓	SC04	Lack of Input Validation	No issues
✓	SC05	Reentrancy Attacks	No issues
✓	SC06	Unchecked External Calls	No issues
✓	SC07	Flash Loan Attacks	No issues
!	SC08	Integer Overflow & Underflow	**4 flagged**
✓	SC09	Insecure Randomness	No issues
✓	SC10	Denial of Service	No issues

SWC Registry

11/12 clear

✓	SWC-107	Reentrancy	No issues
✓	SWC-105	Unprotected access / privileged funcs	No issues
✓	SWC-106	Unprotected selfdestruct	No issues
✓	SWC-115	Authorization via tx.origin	No issues
!	SWC-101	Integer overflow / underflow	**4 flagged**
✓	SWC-104	Unchecked call return value	No issues
✓	SWC-112	Delegatecall to untrusted callee	No issues
✓	SWC-120	Weak randomness	No issues
✓	SWC-116	Block-values as time proxy	No issues
✓	SWC-128	DoS by gas / unbounded ops	No issues
✓	SWC-103	Floating pragma	No issues
✓	SWC-119	Variable shadowing	No issues

DETECTORS EXECUTED · 20

- Potential reentrancy (state write after external call)
- tx.origin used for authorization
- delegatecall to a potentially user-controlled address
- Return value of low-level call not checked
- selfdestruct callable without access control
- Weak randomness from block/tx properties
- block.timestamp dependence
- Privileged function without access control
- Owner holds powerful privileges (centralization)
- Floating pragma
- Arithmetic without overflow protection
- unchecked arithmetic block
- Missing zero-address validation
- Sensitive state change without event
- Shadowed variable declarations
- Loop over dynamic data (gas / DoS)
- Array length read in loop condition
- Post-increment in loop
- Use custom errors instead of require strings
- Public function never called internally

Findings Summary

#	SEVERITY	FINDING	STANDARDS	LOCATION
01	INFORMATIONAL	**unchecked{} block** Arithmetic · low confidence	SWC-101 · SC08	`MMSCash.sol:475`
02	INFORMATIONAL	**unchecked{} block** Arithmetic · low confidence	SWC-101 · SC08	`MMSCash.sol:482`
03	INFORMATIONAL	**unchecked{} block** Arithmetic · low confidence	SWC-101 · SC08	`MMSCash.sol:487`
04	INFORMATIONAL	**unchecked{} block** Arithmetic · low confidence	SWC-101 · SC08	`MMSCash.sol:590`

Gas Optimizations

#	SEVERITY	FINDING	STANDARDS	LOCATION
01	GAS	**`name` can be external** Gas Optimization · low confidence	—	`MMSCash.sol:342`
02	GAS	**`symbol` can be external** Gas Optimization · low confidence	—	`MMSCash.sol:350`
03	GAS	**`decimals` can be external** Gas Optimization · low confidence	—	`MMSCash.sol:367`
04	GAS	**`totalSupply` can be external** Gas Optimization · low confidence	—	`MMSCash.sol:372`
05	GAS	**`balanceOf` can be external** Gas Optimization · low confidence	—	`MMSCash.sol:377`
06	GAS	**`transfer` can be external** Gas Optimization · low confidence	—	`MMSCash.sol:389`
07	GAS	**`approve` can be external** Gas Optimization · low confidence	—	`MMSCash.sol:410`
08	GAS	**`transferFrom` can be external** Gas Optimization · low confidence	—	`MMSCash.sol:432`

DETAILED FINDINGS

Detailed Findings

01 unchecked{} block

INFORMATIONAL SWC-101

SEVERITY
Informational

CONFIDENCE
low

ATTACK PATH
Crafted amount → Arithmetic → **Deviation**

LOCATION
MMSCash.sol
L475-478 / ~621

SWC-101 SWC **SC08** OWASP · Overflow

Detected by: **STATIC**

```
        }
        unchecked {
            // Overflow not possible: value <= fromBalance <= totalSupply.
            _balances[from] = fromBalance - value;
        }
    }
```

Description. An `unchecked` block disables overflow/underflow protection for the enclosed arithmetic.

Impact. If bounds are not otherwise guaranteed, arithmetic here may silently wrap around.

Recommendation. Confirm operands cannot overflow within the unchecked block; document the invariant.

02 unchecked{} block

INFORMATIONAL SWC-101

SEVERITY
Informational

CONFIDENCE
low

ATTACK PATH
Crafted amount → Arithmetic → **Deviation**

LOCATION
MMSCash.sol
L482-485 / ~621

SWC-101 SWC **SC08** OWASP · Overflow

Detected by: **STATIC**

```
        if (to == address(0)) {
            unchecked {
                // Overflow not possible: value <= totalSupply or value <= fromBalance <= totalSupply.
                _totalSupply -= value;
            }
        } else {
```

Description. An `unchecked` block disables overflow/underflow protection for the enclosed arithmetic.

Impact. If bounds are not otherwise guaranteed, arithmetic here may silently wrap around.

Recommendation. Confirm operands cannot overflow within the unchecked block; document the invariant.

03 unchecked{} block

INFORMATIONAL SWC-101

SEVERITY
Informational

CONFIDENCE
low

ATTACK PATH
Crafted amount → Arithmetic → **Deviation**

LOCATION
MMSCash.sol
L487-490 / ~621

SWC-101 SWC **SC08** OWASP · Overflow

Detected by: **STATIC**

```
        } else {
            unchecked {
                // Overflow not possible: balance + value is at most totalSupply, which we know fits into a
                _balances[to] += value;
            }
        }
```

Description. An `unchecked` block disables overflow/underflow protection for the enclosed arithmetic.

Impact. If bounds are not otherwise guaranteed, arithmetic here may silently wrap around.

Recommendation. Confirm operands cannot overflow within the unchecked block; document the invariant.

04 unchecked{} block

INFORMATIONAL SWC-101

SEVERITY
Informational

CONFIDENCE
low

ATTACK PATH
Crafted amount → Arithmetic → **Deviation**

LOCATION
MMSCash.sol
L590-592 / ~621

SWC-101 SWC **SC08** OWASP · Overflow

Detected by: **STATIC**

```
            }
            unchecked {
                _approve(owner, spender, currentAllowance - value, false);
            }
        }
```

Description. An `unchecked` block disables overflow/underflow protection for the enclosed arithmetic.

Impact. If bounds are not otherwise guaranteed, arithmetic here may silently wrap around.

Recommendation. Confirm operands cannot overflow within the unchecked block; document the invariant.

Gas Optimizations

01 `name` can be external

GAS

SEVERITY: Gas

CONFIDENCE: low

ATTACK PATH: Normal call → Gas Optimization → **Higher gas cost**

LOCATION: MMSCash.sol L342-344 / ~621

Detected by: STATIC

```
 */
function name() public view virtual returns (string memory) {
    return _name;
}
```

Description. `name` is `public` but appears to never be called internally.

Impact. Public functions copy arguments to memory; external is cheaper for calldata-only use.

Recommendation. Mark it `external` if it is not called from within the contract.

02 `symbol` can be external

GAS

SEVERITY: Gas

CONFIDENCE: low

ATTACK PATH: Normal call → Gas Optimization → **Higher gas cost**

LOCATION: MMSCash.sol L350-352 / ~621

Detected by: STATIC

```
 */
function symbol() public view virtual returns (string memory) {
    return _symbol;
}
```

Description. `symbol` is `public` but appears to never be called internally.

Impact. Public functions copy arguments to memory; external is cheaper for calldata-only use.

Recommendation. Mark it `external` if it is not called from within the contract.

03 `decimals` can be external

GAS

SEVERITY
Gas

CONFIDENCE
low

ATTACK PATH
Normal call → Gas Optimization →
Higher gas cost

LOCATION
MMSCash.sol
L367-369 / ~621

Detected by: STATIC

```
 */
function decimals() public view virtual returns (uint8) {
    return 18;
}
```

Description. `decimals` is `public` but appears to never be called internally.

Impact. Public functions copy arguments to memory; external is cheaper for calldata-only use.

Recommendation. Mark it `external` if it is not called from within the contract.

04 `totalSupply` can be external

GAS

SEVERITY
Gas

CONFIDENCE
low

ATTACK PATH
Normal call → Gas Optimization →
Higher gas cost

LOCATION
MMSCash.sol
L372-374 / ~621

Detected by: STATIC

```
/// @inheritdoc IERC20
function totalSupply() public view virtual returns (uint256) {
    return _totalSupply;
}
```

Description. `totalSupply` is `public` but appears to never be called internally.

Impact. Public functions copy arguments to memory; external is cheaper for calldata-only use.

Recommendation. Mark it `external` if it is not called from within the contract.

05 `balanceOf` can be external

GAS

SEVERITY
Gas

CONFIDENCE
low

ATTACK PATH
Normal call → Gas Optimization →
Higher gas cost

LOCATION
MMSCash.sol
L377-379 / ~621

Detected by: STATIC

```
    /// @inheritdoc IERC20
    function balanceOf(address account) public view virtual returns (uint256) {
        return _balances[account];
    }
```

Description. `balanceOf` is `public` but appears to never be called internally.

Impact. Public functions copy arguments to memory; external is cheaper for calldata-only use.

Recommendation. Mark it `external` if it is not called from within the contract.

06 `transfer` can be external

GAS

SEVERITY
Gas

CONFIDENCE
low

ATTACK PATH
Normal call → Gas Optimization →
Higher gas cost

LOCATION
MMSCash.sol
L389-393 / ~621

Detected by: STATIC

```
     */
    function transfer(address to, uint256 value) public virtual returns (bool) {
        address owner = _msgSender();
        _transfer(owner, to, value);
        return true;
    }
```

Description. `transfer` is `public` but appears to never be called internally.

Impact. Public functions copy arguments to memory; external is cheaper for calldata-only use.

Recommendation. Mark it `external` if it is not called from within the contract.

07 `approve` can be external

GAS

SEVERITY: Gas

CONFIDENCE: low

ATTACK PATH: Normal call → Gas Optimization → **Higher gas cost**

LOCATION: MMSCash.sol L410-414 / ~621

Detected by: STATIC

```
 */
function approve(address spender, uint256 value) public virtual returns (bool) {
    address owner = _msgSender();
    _approve(owner, spender, value);
    return true;
}
```

Description. `approve` is `public` but appears to never be called internally.

Impact. Public functions copy arguments to memory; external is cheaper for calldata-only use.

Recommendation. Mark it `external` if it is not called from within the contract.

08 `transferFrom` can be external

GAS

SEVERITY: Gas

CONFIDENCE: low

ATTACK PATH: Normal call → Gas Optimization → **Higher gas cost**

LOCATION: MMSCash.sol L432-437 / ~621

Detected by: STATIC

```
 */
function transferFrom(address from, address to, uint256 value) public virtual returns (bool) {
    address spender = _msgSender();
    _spendAllowance(from, spender, value);
    _transfer(from, to, value);
    return true;
}
```

Description. `transferFrom` is `public` but appears to never be called internally.

Impact. Public functions copy arguments to memory; external is cheaper for calldata-only use.

Recommendation. Mark it `external` if it is not called from within the contract.

Appendix

Severity Classification

SEVERITY	DESCRIPTION
CRITICAL	Directly exploitable; leads to loss of funds or renders the contract unusable.
HIGH	Serious vulnerability that can be exploited under realistic conditions.
MEDIUM	Exploitable under specific conditions or with limited impact.
LOW	Minor issue or deviation from best practice with low impact.
INFORMATIONAL	Observation or note; no direct security impact.
GAS	Opportunity to reduce gas consumption.

Disclaimer. This report is provided for informational purposes only and does not constitute a warranty regarding the security of the audited code. Automated and AI-assisted analysis cannot guarantee the absence of vulnerabilities. A manual review by qualified auditors is recommended before deploying to production.