CACTUS

Smart Contract Audit Report

SYSFIXED

TABLE OF CONTENTS

SYSFIXED

AUDITED DETAILS

Audited Project

Project name	Token ticker	Blockchain
CACTUS	CACTUS	Binance Smart Chain

Addresses

Contract address	0x6034120575f3a4FDC03196128f4AD2011D570A96
Contract deployer address	0x8ACf11258D871276973a9933f985a400FeDe72B7

Project Website

https://cactustoken.com/

Codebase

https://bscscan.com/address/0x6034120575f3a4FDC03196128f4AD2011D570A96#code

SUMMARY

CACTUS is a deflationary community token that equally rewards every participant in its ecosystem. Hold CACTUS & Earn BUSD! | 3% to Holders, 2% to Liquidity & 3% to Ecosystem.

Contract Summary

Documentation Quality

CACTUS provides a very good documentation with standard of solidity base code.

- The technical description is provided clearly and structured and also dont have any high risk issue.

Code Quality

The Overall quality of the basecode is standard.

- Standard solidity basecode and rules are already followed by CACTUS with the discovery of several low issues.

Test Coverage

Test coverage of the project is 100% (Through Codebase)

Audit Findings Summary

- SWC-101 | It is recommended to use vetted safe math libraries for arithmetic operations consistently on lines 305, 324, 346, 379, 381, 402, 403, 428, 430, 601, 615, 630, 631, 644, 656, 671, 685, 699, 713, 729, 752, 775, 801, 1720, 1739, 1761, 1794, 1796, 1817, 1818, 1843, 1845, 2072, 2076, 2088, 2095, 2104, 2205, 2309, 2344, 2431, 2716, 2726, 2730, 2937, 2937, 2957 and 2205.
- SWC-110 SWC-123 | It is recommended to use of revert(), assert(), and require() in Solidity, and the new REVERT opcode in the EVM on lines 2174, 2206, 2211, 2722, 2875, 2876, 2886, 2887, 2888, 2897, 2904, 2958, 3267, 3268, 3284, 3285 and 3286.
- SWC-115 | tx.origin should not be used for authorization, use msg.sender instead on lines 3123 and 3223.

CONCLUSION

We have audited the CACTUS project released on January -2023 to discover issues and identify potential security vulnerabilities in CACTUS Project. This process is used to find technical issues and security loopholes which might be found in the smart contract.

The security audit report provides a satisfactory result with some low-risk issues.

The issues found in the CACTUS smart contract code do not pose a considerable risk. The writing of the contract is close to the standard of writing contracts in general. The low-risk issues found are some arithmetic operation issues, tx.origin as a part of authorization control and out of bounds array access which the index access expression can cause an exception in case of the use of an invalid array index value. We recommend avoiding "tx.origin" using "tx.origin" as a security control can lead to authorization bypass vulnerabilities. Consider using "msg.sender" unless you really know what you are doing.

AUDIT RESULT

Article	Category	Description	Result
Default Visibility	**SWC-100** **SWC-108**	Functions and state variables visibility should be set explicitly. Visibility levels should be specified consciously.	**PASS**
Integer Overflow and Underflow	**SWC-101**	If unchecked math is used, all math operations should be safe from overflows and underflows.	**ISSUE FOUND**
Outdated Compiler Version	**SWC-102**	It is recommended to use a recent version of the Solidity compiler.	**PASS**
Floating Pragma	**SWC-103**	Contracts should be deployed with the same compiler version and flags that they have been tested thoroughly.	**PASS**
Unchecked Call Return Value	**SWC-104**	The return value of a message call should be checked.	**PASS**
Unprotected Ether Withdrawal	**SWC-105**	Due to missing or insufficient access controls, malicious parties can withdraw from the contract.	**PASS**
SELFDESTRUCT Instruction	**SWC-106**	The contract should not be self-destructible while it has funds belonging to users.	**PASS**
Reentrancy	**SWC-107**	Check effect interaction pattern should be followed if the code performs recursive call.	**PASS**
Uninitialized Storage Pointer	**SWC-109**	Uninitialized local storage variables can point to unexpected storage locations in the contract.	**PASS**
Assert Violation	**SWC-110** **SWC-123**	Properly functioning code should never reach a failing assert statement.	**ISSUE FOUND**
Deprecated Solidity Functions	**SWC-111**	Deprecated built-in functions should never be used.	**PASS**
Delegate call to Untrusted Callee	**SWC-112**	Delegatecalls should only be allowed to trusted addresses.	**PASS**

DoS (Denial of Service)	SWC-113 SWC-128	Execution of the code should never be blocked by a specific contract state unless required.	PASS
Race Conditions	SWC-114	Race Conditions and Transactions Order Dependency should not be possible.	PASS
Authorization through tx.origin	SWC-115	tx.origin should not be used for authorization.	ISSUE FOUND
Block values as a proxy for time	SWC-116	Block numbers should not be used for time calculations.	PASS
Signature Unique ID	SWC-117 SWC-121 SWC-122	Signed messages should always have a unique id. A transaction hash should not be used as a unique id.	PASS
Incorrect Constructor Name	SWC-118	Constructors are special functions that are called only once during the contract creation.	PASS
Shadowing State Variable	SWC-119	State variables should not be shadowed.	PASS
Weak Sources of Randomness	SWC-120	Random values should never be generated from Chain Attributes or be predictable.	PASS
Write to Arbitrary Storage Location	SWC-124	The contract is responsible for ensuring that only authorized user or contract accounts may write to sensitive storage locations.	PASS
Incorrect Inheritance Order	SWC-125	When inheriting multiple contracts, especially if they have identical functions, a developer should carefully specify inheritance in the correct order. The rule of thumb is to inherit contracts from more /general/ to more /specific/.	PASS
Insufficient Gas Griefing	SWC-126	Insufficient gas griefing attacks can be performed on contracts which accept data and use it in a sub-call on another contract.	PASS
Arbitrary Jump Function	SWC-127	As Solidity doesnt support pointer arithmetics, it is impossible to change such variable to an arbitrary value.	PASS

Typographical Error	SWC-129	A typographical error can occur for example when the intent of a defined operation is to sum a number to a variable.	PASS
Override control character	SWC-130	Malicious actors can use the Right-To-Left-Override unicode character to force RTL text rendering and confuse users as to the real intent of a contract.	PASS
Unused variables	SWC-131 SWC-135	Unused variables are allowed in Solidity and they do not pose a direct security issue.	PASS
Unexpected Ether balance	SWC-132	Contracts can behave erroneously when they strictly assume a specific Ether balance.	PASS
Hash Collisions Variable	SWC-133	Using abi.encodePacked() with multiple variable length arguments can, in certain situations, lead to a hash collision.	PASS
Hardcoded gas amount	SWC-134	The transfer() and send() functions forward a fixed amount of 2300 gas.	PASS
Unencrypted Private Data	SWC-136	It is a common misconception that private type variables cannot be read.	PASS

SMART CONTRACT ANALYSIS

Started	Wednesday Jan 04 2023 17:32:53 GMT+0000 (Coordinated Universal Time)
Finished	Thursday Jan 05 2023 16:40:27 GMT+0000 (Coordinated Universal Time)
Mode	Standard
Main Source File	BABYTOKEN.sol

Detected Issues

ID	Title	Severity	Status
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged

SWC-101	ARITHMETIC OPERATION "%" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "%" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "%" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-=" DISCOVERED	low	acknowledged

SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "**" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "++" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "++" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "++" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "**" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "++" DISCOVERED	low	acknowledged
SWC-101	COMPILER-REWRITABLE "<UINT> - 1" DISCOVERED	low	acknowledged
SWC-115	USE OF "TX.ORIGIN" AS A PART OF AUTHORIZATION CONTROL.	low	acknowledged
SWC-115	USE OF "TX.ORIGIN" AS A PART OF AUTHORIZATION CONTROL.	low	acknowledged

SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED
LINE 305

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
304    unchecked {
305    _approve(sender, _msgSender(), currentAllowance - amount);
306    }
307
308    return true;
309
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 324

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
323    function increaseAllowance(address spender, uint256 addedValue) public virtual
returns (bool) {
324    _approve(_msgSender(), spender, _allowances[_msgSender()][spender] + addedValue);
325    return true;
326    }
327
328
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED
LINE 346

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
345    unchecked {
346    _approve(_msgSender(), spender, currentAllowance - subtractedValue);
347    }
348
349    return true;
350
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED
LINE 379

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
378    unchecked {
379    _balances[sender] = senderBalance - amount;
380    }
381    _balances[recipient] += amount;
382
383
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED
LINE 381

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
380    }
381    _balances[recipient] += amount;
382
383    emit Transfer(sender, recipient, amount);
384
385
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED
LINE 402

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
401
402   _totalSupply += amount;
403   _balances[account] += amount;
404   emit Transfer(address(0), account, amount);
405
406
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED
LINE 403

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
402    _totalSupply += amount;
403    _balances[account] += amount;
404    emit Transfer(address(0), account, amount);
405
406    _afterTokenTransfer(address(0), account, amount);
407
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED
LINE 428

low SEVERITY
This plugin produces issues to support false positive discovery within mythril.

Source File
- BABYTOKEN.sol

Locations

```
427    unchecked {
428    _balances[account] = accountBalance - amount;
429    }
430    _totalSupply -= amount;
431
432
```

SWC-101 | ARITHMETIC OPERATION "-=" DISCOVERED
LINE 430

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
429    }
430    _totalSupply -= amount;
431
432    emit Transfer(account, address(0), amount);
433
434
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 601

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
600    unchecked {
601    uint256 c = a + b;
602    if (c < a) return (false, 0);
603    return (true, c);
604    }
605
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED
LINE 615

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
614    if (b > a) return (false, 0);
615    return (true, a - b);
616    }
617    }
618
619
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED
LINE 630

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
629   if (a == 0) return (true, 0);
630   uint256 c = a * b;
631   if (c / a != b) return (false, 0);
632   return (true, c);
633   }
634
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED
LINE 631

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
630   uint256 c = a * b;
631   if (c / a != b) return (false, 0);
632   return (true, c);
633   }
634   }
635
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED
LINE 644

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
643   if (b == 0) return (false, 0);
644   return (true, a / b);
645   }
646   }
647
648
```

SYSFIXED

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED
LINE 656

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
655   if (b == 0) return (false, 0);
656   return (true, a % b);
657   }
658   }
659
660
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED
LINE 671

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
670    function add(uint256 a, uint256 b) internal pure returns (uint256) {
671    return a + b;
672    }
673
674    /**
675
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED
LINE 685

low SEVERITY
This plugin produces issues to support false positive discovery within mythril.

Source File
- BABYTOKEN.sol

Locations

```
684    function sub(uint256 a, uint256 b) internal pure returns (uint256) {
685    return a - b;
686    }
687
688    /**
689
```

SYSFIXED

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED
LINE 699

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
698    function mul(uint256 a, uint256 b) internal pure returns (uint256) {
699    return a * b;
700    }
701
702    /**
703
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED
LINE 713

low SEVERITY
This plugin produces issues to support false positive discovery within mythril.

Source File
- BABYTOKEN.sol

Locations

```
712    function div(uint256 a, uint256 b) internal pure returns (uint256) {
713    return a / b;
714    }
715
716    /**
717
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 729

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
728    function mod(uint256 a, uint256 b) internal pure returns (uint256) {
729    return a % b;
730    }
731
732    /**
733
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED
LINE 752

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
751    require(b <= a, errorMessage);
752    return a - b;
753    }
754    }
755
756
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED
LINE 775

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
774    require(b > 0, errorMessage);
775    return a / b;
776    }
777    }
778
779
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED
LINE 801

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
800    require(b > 0, errorMessage);
801    return a % b;
802    }
803    }
804    }
805
```

SYSFIXED

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED
LINE 1720

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
1719   unchecked {
1720   _approve(sender, _msgSender(), currentAllowance - amount);
1721   }
1722
1723   return true;
1724
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED
LINE 1739

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
1738    function increaseAllowance(address spender, uint256 addedValue) public virtual
returns (bool) {
1739    _approve(_msgSender(), spender, _allowances[_msgSender()][spender] + addedValue);
1740    return true;
1741    }
1742
1743
```

SYSFIXED

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED
LINE 1761

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
1760   unchecked {
1761   _approve(_msgSender(), spender, currentAllowance - subtractedValue);
1762   }
1763
1764   return true;
1765
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 1794

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
1793   unchecked {
1794   _balances[sender] = senderBalance - amount;
1795   }
1796   _balances[recipient] += amount;
1797
1798
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED
LINE 1796

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
1795    }
1796    _balances[recipient] += amount;
1797
1798    emit Transfer(sender, recipient, amount);
1799
1800
```

SYSFIXED

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED
LINE 1817

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
1816
1817    _totalSupply += amount;
1818    _balances[account] += amount;
1819    emit Transfer(address(0), account, amount);
1820
1821
```

SYSFIXED

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED
LINE 1818

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
1817    _totalSupply += amount;
1818    _balances[account] += amount;
1819    emit Transfer(address(0), account, amount);
1820
1821    _afterTokenTransfer(address(0), account, amount);
1822
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED
LINE 1843

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
1842    unchecked {
1843    _balances[account] = accountBalance - amount;
1844    }
1845    _totalSupply -= amount;
1846
1847
```

SWC-101 | ARITHMETIC OPERATION "-=" DISCOVERED
LINE 1845

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
1844    }
1845    _totalSupply -= amount;
1846
1847    emit Transfer(account, address(0), amount);
1848
1849
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED
LINE 2072

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
2071    function mul(int256 a, int256 b) internal pure returns (int256) {
2072    int256 c = a * b;
2073
2074    // Detect overflow when multiplying MIN_INT256 with -1
2075    require(c != MIN_INT256 || (a & MIN_INT256) != (b & MIN_INT256));
2076
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED
LINE 2076

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
2075   require(c != MIN_INT256 || (a & MIN_INT256) != (b & MIN_INT256));
2076   require((b == 0) || (c / b == a));
2077   return c;
2078   }
2079
2080
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED
LINE 2088

low SEVERITY
This plugin produces issues to support false positive discovery within mythril.

Source File
- BABYTOKEN.sol

Locations

```
2087    // Solidity already throws when dividing by 0.
2088    return a / b;
2089    }
2090
2091    /**
2092
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED
LINE 2095

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
2094   function sub(int256 a, int256 b) internal pure returns (int256) {
2095   int256 c = a - b;
2096   require((b >= 0 && c <= a) || (b < 0 && c > a));
2097   return c;
2098   }
2099
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 2104

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
2103    function add(int256 a, int256 b) internal pure returns (int256) {
2104    int256 c = a + b;
2105    require((b >= 0 && c >= a) || (b < 0 && c < a));
2106    return c;
2107    }
2108
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED
LINE 2205

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
2204   uint256 index = map.indexOf[key];
2205   uint256 lastIndex = map.keys.length - 1;
2206   address lastKey = map.keys[lastIndex];
2207
2208   map.indexOf[lastKey] = index;
2209
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 2309

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
2308    //  see https://github.com/ethereum/EIPs/issues/1726#issuecomment-472352728
2309    uint256 internal constant magnitude = 2**128;
2310
2311    uint256 internal magnifiedDividendPerShare;
2312
2313
```

SYSFIXED

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED
LINE 2344

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
2343    magnifiedDividendPerShare = magnifiedDividendPerShare.add(
2344    (amount).mul(magnitude) / totalSupply()
2345    );
2346    emit DividendsDistributed(msg.sender, amount);
2347
2348
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED
LINE 2431

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
2430   return
2431   magnifiedDividendPerShare
2432   .mul(balanceOf(_owner))
2433   .toInt256Safe()
2434   .add(magnifiedDividendCorrections[_owner])
2435
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED
LINE 2716

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
2715    while (gasUsed < gas && iterations < numberOfTokenHolders) {
2716    _lastProcessedIndex++;
2717
2718    if (_lastProcessedIndex >= tokenHoldersMap.keys.length) {
2719    _lastProcessedIndex = 0;
2720
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED
LINE 2726

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
2725    if (processAccount(payable(account), true)) {
2726    claims++;
2727    }
2728    }
2729
2730
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED
LINE 2730

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
2729
2730    iterations++;
2731
2732    uint256 newGasLeft = gasleft();
2733
2734
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED
LINE 2937

low SEVERITY
This plugin produces issues to support false positive discovery within mythril.

Source File
- BABYTOKEN.sol

Locations

```
2936   require(
2937   amount > totalSupply() / 10**5,
2938   "BABYTOKEN: Amount must be greater than 0.001% of total supply"
2939   );
2940   swapTokensAtAmount = amount;
2941
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED
LINE 2937

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
2936   require(
2937   amount > totalSupply() / 10**5,
2938   "BABYTOKEN: Amount must be greater than 0.001% of total supply"
2939   );
2940   swapTokensAtAmount = amount;
2941
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED
LINE 2957

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
2956    {
2957    for (uint256 i = 0; i < accounts.length; i++) {
2958    _isExcludedFromFees[accounts[i]] = true;
2959    }
2960
2961
```

SWC-101 | COMPILER-REWRITABLE "<UINT> - 1" DISCOVERED
LINE 2205

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BABYTOKEN.sol

Locations

```
2204   uint256 index = map.indexOf[key];
2205   uint256 lastIndex = map.keys.length - 1;
2206   address lastKey = map.keys[lastIndex];
2207
2208   map.indexOf[lastKey] = index;
2209
```

SYSFIXED

SWC-115 | USE OF "TX.ORIGIN" AS A PART OF AUTHORIZATION CONTROL.

LINE 3123

low SEVERITY

Using "tx.origin" as a security control can lead to authorization bypass vulnerabilities. Consider using "msg.sender" unless you really know what you are doing.

Source File

- BABYTOKEN.sol

Locations

```
3122   gas,
3123   tx.origin
3124   );
3125   }
3126
3127
```

SYSFIXED

SWC-115 | USE OF "TX.ORIGIN" AS A PART OF AUTHORIZATION CONTROL.

LINE 3223

low SEVERITY

Using "tx.origin" as a security control can lead to authorization bypass vulnerabilities. Consider using "msg.sender" unless you really know what you are doing.

Source File

- BABYTOKEN.sol

Locations

```
3222   gas,
3223   tx.origin
3224   );
3225   } catch {}
3226   }
3227
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS
LINE 2174

low SEVERITY

The index access expression can cause an exception in case of use of invalid array index value.

Source File

- BABYTOKEN.sol

Locations

```
2173    {
2174    return map.keys[index];
2175    }
2176
2177    function size(Map storage map) public view returns (uint256) {
2178
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS
LINE 2206

low SEVERITY

The index access expression can cause an exception in case of use of invalid array index value.

Source File

- BABYTOKEN.sol

Locations

```
2205    uint256 lastIndex = map.keys.length - 1;
2206    address lastKey = map.keys[lastIndex];
2207
2208    map.indexOf[lastKey] = index;
2209    delete map.indexOf[key];
2210
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS
LINE 2211

low SEVERITY

The index access expression can cause an exception in case of use of invalid array index value.

Source File

- BABYTOKEN.sol

Locations

```
2210
2211   map.keys[index] = lastKey;
2212   map.keys.pop();
2213   }
2214   }
2215
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS
LINE 2722

low SEVERITY

The index access expression can cause an exception in case of use of invalid array index value.

Source File

- BABYTOKEN.sol

Locations

```
2721
2722    address account = tokenHoldersMap.keys[_lastProcessedIndex];
2723
2724    if (canAutoClaim(lastClaimTimes[account])) {
2725    if (processAccount(payable(account), true)) {
2726
```

SYSFIXED

SWC-110 | OUT OF BOUNDS ARRAY ACCESS
LINE 2875

low SEVERITY

The index access expression can cause an exception in case of use of invalid array index value.

Source File

- BABYTOKEN.sol

Locations

```
2874   ) payable ERC20(name_, symbol_) {
2875   rewardToken = addrs[0];
2876   _marketingWalletAddress = addrs[2];
2877   require(
2878   msg.sender != _marketingWalletAddress,
2879
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS
LINE 2876

low SEVERITY

The index access expression can cause an exception in case of use of invalid array index value.

Source File

- BABYTOKEN.sol

Locations

```
2875    rewardToken = addrs[0];
2876    _marketingWalletAddress = addrs[2];
2877    require(
2878    msg.sender != _marketingWalletAddress,
2879    "Owner and marketing wallet cannot be the same"
2880
```

SYSFIXED

SWC-110 | OUT OF BOUNDS ARRAY ACCESS
LINE 2886

low SEVERITY

The index access expression can cause an exception in case of use of invalid array index value.

Source File

- BABYTOKEN.sol

Locations

```
2885
2886    tokenRewardsFee = feeSettings[0];
2887    liquidityFee = feeSettings[1];
2888    marketingFee = feeSettings[2];
2889    totalFees = tokenRewardsFee.add(liquidityFee).add(marketingFee);
2890
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS
LINE 2887

low SEVERITY

The index access expression can cause an exception in case of use of invalid array index value.

Source File

- BABYTOKEN.sol

Locations

```
2886    tokenRewardsFee = feeSettings[0];
2887    liquidityFee = feeSettings[1];
2888    marketingFee = feeSettings[2];
2889    totalFees = tokenRewardsFee.add(liquidityFee).add(marketingFee);
2890    require(totalFees <= 25, "Total fee is over 25%");
2891
```

SYSFIXED

SWC-110 | OUT OF BOUNDS ARRAY ACCESS
LINE 2888

low SEVERITY

The index access expression can cause an exception in case of use of invalid array index value.

Source File

- BABYTOKEN.sol

Locations

```
2887    liquidityFee = feeSettings[1];
2888    marketingFee = feeSettings[2];
2889    totalFees = tokenRewardsFee.add(liquidityFee).add(marketingFee);
2890    require(totalFees <= 25, "Total fee is over 25%");
2891    swapTokensAtAmount = totalSupply_.div(1000); // 0.1%
2892
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS
LINE 2897

low SEVERITY
The index access expression can cause an exception in case of use of invalid array index value.

Source File
- BABYTOKEN.sol

Locations

```
2896    dividendTracker = BABYTOKENDividendTracker(
2897    payable(Clones.clone(addrs[3]))
2898    );
2899    dividendTracker.initialize(
2900    rewardToken,
2901
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS
LINE 2904

low SEVERITY

The index access expression can cause an exception in case of use of invalid array index value.

Source File

- BABYTOKEN.sol

Locations

```
2903
2904    IUniswapV2Router02 _uniswapV2Router = IUniswapV2Router02(addrs[1]);
2905    // Create a uniswap pair for this new token
2906    address _uniswapV2Pair = IUniswapV2Factory(_uniswapV2Router.factory())
2907    .createPair(address(this), _uniswapV2Router.WETH());
2908
```

SYSFIXED

SWC-110 | OUT OF BOUNDS ARRAY ACCESS
LINE 2958

low SEVERITY

The index access expression can cause an exception in case of use of invalid array index value.

Source File

- BABYTOKEN.sol

Locations

```
2957   for (uint256 i = 0; i < accounts.length; i++) {
2958   _isExcludedFromFees[accounts[i]] = true;
2959   }
2960
2961   emit ExcludeMultipleAccountsFromFees(accounts);
2962
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS
LINE 3267

low SEVERITY

The index access expression can cause an exception in case of use of invalid array index value.

Source File

- BABYTOKEN.sol

Locations

```
3266   address[] memory path = new address[](2);
3267   path[0] = address(this);
3268   path[1] = uniswapV2Router.WETH();
3269
3270   _approve(address(this), address(uniswapV2Router), tokenAmount);
3271
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS
LINE 3268

low SEVERITY

The index access expression can cause an exception in case of use of invalid array index value.

Source File

- BABYTOKEN.sol

Locations

```
3267   path[0] = address(this);
3268   path[1] = uniswapV2Router.WETH();
3269
3270   _approve(address(this), address(uniswapV2Router), tokenAmount);
3271
3272
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS
LINE 3284

low SEVERITY

The index access expression can cause an exception in case of use of invalid array index value.

Source File

- BABYTOKEN.sol

Locations

```
3283    address[] memory path = new address[](3);
3284    path[0] = address(this);
3285    path[1] = uniswapV2Router.WETH();
3286    path[2] = rewardToken;
3287
3288
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS
LINE 3285

low SEVERITY

The index access expression can cause an exception in case of use of invalid array index value.

Source File

- BABYTOKEN.sol

Locations

```
3284   path[0] = address(this);
3285   path[1] = uniswapV2Router.WETH();
3286   path[2] = rewardToken;
3287
3288   _approve(address(this), address(uniswapV2Router), tokenAmount);
3289
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS
LINE 3286

low SEVERITY

The index access expression can cause an exception in case of use of invalid array index value.

Source File

- BABYTOKEN.sol

Locations

```
3285   path[1] = uniswapV2Router.WETH();
3286   path[2] = rewardToken;
3287
3288   _approve(address(this), address(uniswapV2Router), tokenAmount);
3289
3290
```

DISCLAIMER

ABOUT US

Sysfixed is a blockchain security certification organization established in 2021 with the objective to provide smart contract security services and verify their correctness in blockchain-based protocols. Sysfixed automatically scans for security vulnerabilities in Ethereum and other EVM-based blockchain smart contracts. Sysfixed a comprehensive range of analysis techniques—including static analysis, dynamic analysis, and symbolic execution—can accurately detect security vulnerabilities to provide an in-depth analysis report. With a vibrant ecosystem of world-class integration partners that amplify developer productivity, Sysfixed can be utilized in all phases of your project's lifecycle. Our team of security experts is dedicated to the research and improvement of our tools and techniques used to fortify your code.