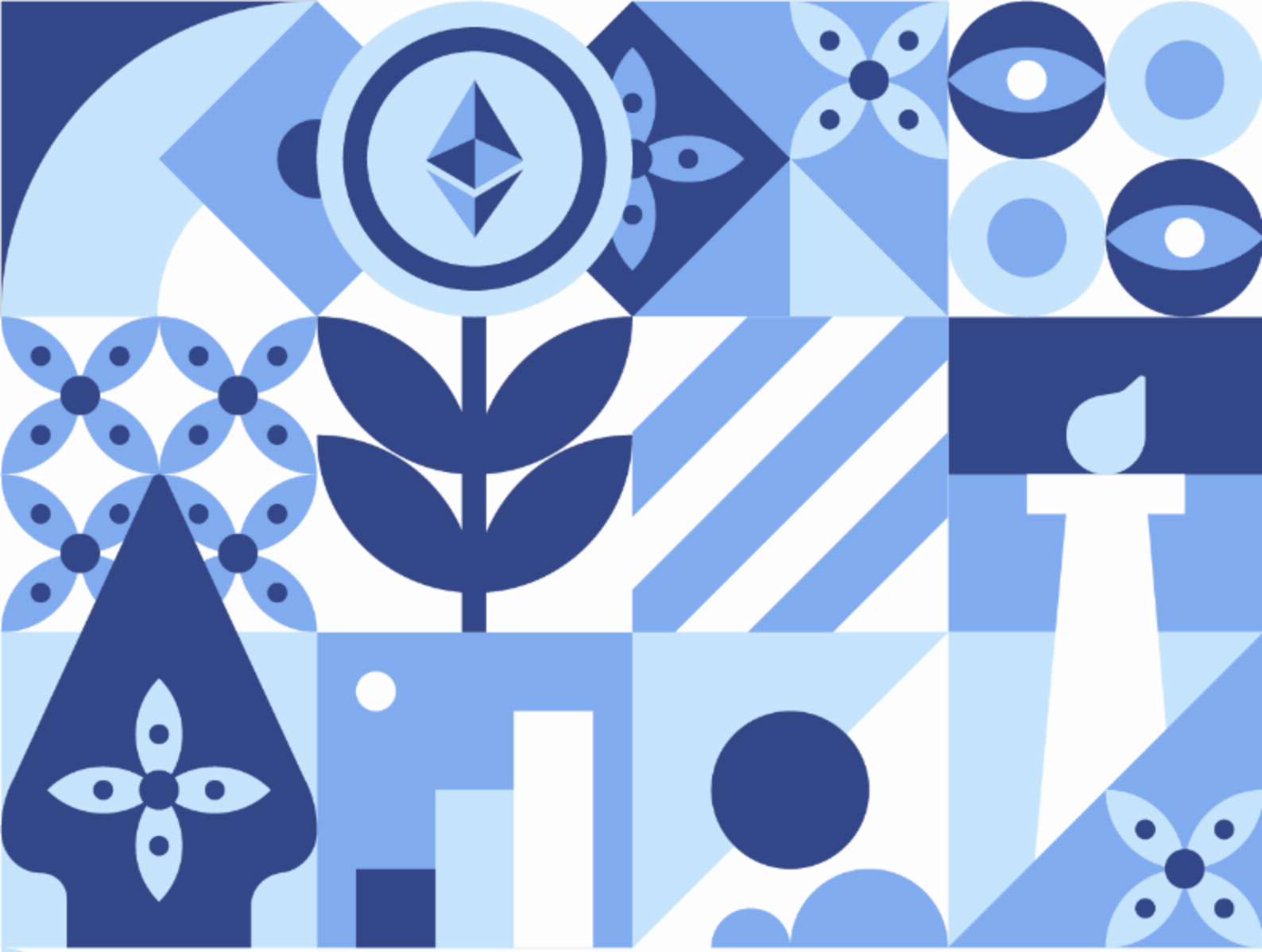

BUY BACK

Smart Contract Audit Report

SYSFIXED

28 Jan 2023

TABLE OF CONTENTS

Audited Details

- Audited Project
- Blockchain
- Addresses
- Project Website
- Codebase

Summary

- Contract Summary
- Audit Findings Summary
- Vulnerabilities Summary

Conclusion

Audit Results

Smart Contract Analysis

- Detected Vulnerabilities

Disclaimer

About Us

AUDITED DETAILS

Audited Project

Project name	Token ticker	Blockchain
BUY BACK	$BB	Ethereum

Addresses

Contract address	0x5e8017b3cf82d338703ebb7c8a037dbbd5b2a396
Contract deployer address	0xBd55B07Cc0F8aD4088D3443182b8b00Af2225253

Project Website

https://buybackwallet.com/

Codebase

https://etherscan.io/address/0x5e8017b3cf82d338703ebb7c8a037dbbd5b2a396#code

SUMMARY

BBB is a token to counter all sells . 200 dollar sell will be countered by 201 dollar buy. All info on website.

Contract Summary

Documentation Quality

BUY BACK 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 BUY BACK 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 203, 214, 226, 249, 251, 266, 267, 472, 480, 488, 496, 512, 526, 541, 542, 555, 567, 576, 584, 592, 600, 608, 622, 637, 652, 807, 807, 808, 808, 809, 809, 810, 810, 861, 861, 888, 915, 920, 927, 932, 938, 943, 973 and 978.
- 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 1001 and 1002.

CONCLUSION

We have audited the BUY BACK project released on January 2023 to discover issues and identify potential security vulnerabilities in BUY BACKProject. 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 BUY BACK 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, 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.

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.	PASS
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	Friday Jan 27 2023 04:45:28 GMT+0000 (Coordinated Universal Time)
Finished	Saturday Jan 28 2023 07:10:38 GMT+0000 (Coordinated Universal Time)
Mode	Standard
Main Source File	BB.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-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 203

low SEVERITY

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

Source File

- BB.sol

Locations

```
202   unchecked {
203   _approve(sender, _msgSender(), currentAllowance - amount);
204   }
205
206   return true;
207
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 214

low SEVERITY

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

Source File

- BB.sol

Locations

```
213   function increaseAllowance(address spender, uint256 addedValue) public virtual
returns (bool) {
214   _approve(_msgSender(), spender, _allowances[_msgSender()][spender] + addedValue);
215   return true;
216   }
217
218
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 226

low SEVERITY

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

Source File

- BB.sol

Locations

```
225   unchecked {
226   _approve(_msgSender(), spender, currentAllowance - subtractedValue);
227   }
228
229   return true;
230
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 249

low SEVERITY

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

Source File

- BB.sol

Locations

```
248   unchecked {
249   _balances[sender] = senderBalance - amount;
250   }
251   _balances[recipient] += amount;
252
253
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 251

low SEVERITY

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

Source File

- BB.sol

Locations

```
250   }
251   _balances[recipient] += amount;
252
253   emit Transfer(sender, recipient, amount);
254
255
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 266

low SEVERITY

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

Source File

- BB.sol

Locations

```
265
266   _totalSupply += amount;
267   _balances[account] += amount;
268   emit Transfer(address(0), account, amount);
269
270
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 267

low SEVERITY

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

Source File

- BB.sol

Locations

```
266   _totalSupply += amount;
267   _balances[account] += amount;
268   emit Transfer(address(0), account, amount);
269
270   _afterTokenTransfer(address(0), account, amount);
271
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 472

low SEVERITY

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

Source File

- BB.sol

Locations

```
471   function mul(int256 a, int256 b) internal pure returns (int256) {
472   return a * b;
473   }
474
475   /**
476
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 480

low SEVERITY

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

Source File

- BB.sol

Locations

```
479   function div(int256 a, int256 b) internal pure returns (int256) {
480   return a / b;
481   }
482
483   /**
484
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 488

low SEVERITY

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

Source File

- BB.sol

Locations

```
487   function sub(int256 a, int256 b) internal pure returns (int256) {
488   return a - b;
489   }
490
491   /**
492
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 496

low SEVERITY

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

Source File

- BB.sol

Locations

```
495   function add(int256 a, int256 b) internal pure returns (int256) {
496   return a + b;
497   }
498   }
499
500
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 512

low SEVERITY

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

Source File

- BB.sol

Locations

```
511   unchecked {
512   uint256 c = a + b;
513   if (c < a) return (false, 0);
514   return (true, c);
515   }
516
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 526

low SEVERITY

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

Source File

- BB.sol

Locations

```
525   if (b > a) return (false, 0);
526   return (true, a - b);
527   }
528   }
529
530
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 541

low SEVERITY

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

Source File

- BB.sol

Locations

```
540   if (a == 0) return (true, 0);
541   uint256 c = a * b;
542   if (c / a != b) return (false, 0);
543   return (true, c);
544   }
545
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 542

low SEVERITY

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

Source File

- BB.sol

Locations

```
541   uint256 c = a * b;
542   if (c / a != b) return (false, 0);
543   return (true, c);
544   }
545   }
546
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 555

low SEVERITY

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

Source File

- BB.sol

Locations

```
554   if (b == 0) return (false, 0);
555   return (true, a / b);
556   }
557   }
558
559
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 567

low SEVERITY

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

Source File

- BB.sol

Locations

```
566   if (b == 0) return (false, 0);
567   return (true, a % b);
568   }
569   }
570
571
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 576

low SEVERITY

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

Source File

- BB.sol

Locations

```
575   function add(uint256 a, uint256 b) internal pure returns (uint256) {
576   return a + b;
577   }
578
579   /**
580
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 584

low SEVERITY

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

Source File

- BB.sol

Locations

```
583   function sub(uint256 a, uint256 b) internal pure returns (uint256) {
584   return a - b;
585   }
586
587   /**
588
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 592

low SEVERITY

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

Source File

- BB.sol

Locations

```
591   function mul(uint256 a, uint256 b) internal pure returns (uint256) {
592   return a * b;
593   }
594
595   /**
596
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 600

low SEVERITY

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

Source File

- BB.sol

Locations

```
599   function div(uint256 a, uint256 b) internal pure returns (uint256) {
600   return a / b;
601   }
602
603   /**
604
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 608

low SEVERITY

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

Source File

- BB.sol

Locations

```
607   function mod(uint256 a, uint256 b) internal pure returns (uint256) {
608   return a % b;
609   }
610
611   /**
612
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 622

low SEVERITY

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

Source File

- BB.sol

Locations

```
621   require(b <= a, errorMessage);
622   return a - b;
623   }
624   }
625
626
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 637

low SEVERITY

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

Source File

- BB.sol

Locations

```
636   require(b > 0, errorMessage);
637   return a / b;
638   }
639   }
640
641
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 652

low SEVERITY

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

Source File

- BB.sol

Locations

```
651   require(b > 0, errorMessage);
652   return a % b;
653   }
654   }
655   }
656
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 807

low SEVERITY

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

Source File

- BB.sol

Locations

```
806   uint256 public liquidityTransferFee = 8;
807   uint256 public maxBuyTransactionAmount = 220000 * (10**18);
808   uint256 public maxSellTransactionAmount = 220000 * (10**18);
809   uint256 public swapTokensAtAmount = 4000 * (10**18);
810   uint256 public maxWalletToken = 220000 * (10**18);
811
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 807

low SEVERITY

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

Source File

- BB.sol

Locations

```
806   uint256 public liquidityTransferFee = 8;
807   uint256 public maxBuyTransactionAmount = 220000 * (10**18);
808   uint256 public maxSellTransactionAmount = 220000 * (10**18);
809   uint256 public swapTokensAtAmount = 4000 * (10**18);
810   uint256 public maxWalletToken = 220000 * (10**18);
811
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 808

low SEVERITY

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

Source File

- BB.sol

Locations

```
807   uint256 public maxBuyTransactionAmount = 220000 * (10**18);
808   uint256 public maxSellTransactionAmount = 220000 * (10**18);
809   uint256 public swapTokensAtAmount = 4000 * (10**18);
810   uint256 public maxWalletToken = 220000 * (10**18);
811
812
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 808

low SEVERITY

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

Source File

- BB.sol

Locations

```
807   uint256 public maxBuyTransactionAmount = 220000 * (10**18);
808   uint256 public maxSellTransactionAmount = 220000 * (10**18);
809   uint256 public swapTokensAtAmount = 4000 * (10**18);
810   uint256 public maxWalletToken = 220000 * (10**18);
811
812
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 809

low SEVERITY

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

Source File

- BB.sol

Locations

```
808   uint256 public maxSellTransactionAmount = 220000 * (10**18);
809   uint256 public swapTokensAtAmount = 4000 * (10**18);
810   uint256 public maxWalletToken = 220000 * (10**18);
811
812   address payable public buybackWallet =
payable(0xfebB0033c348ce91C19d7B075134ec7A7e044e1b);
813
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 809

low SEVERITY

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

Source File

- BB.sol

Locations

```
808   uint256 public maxSellTransactionAmount = 220000 * (10**18);
809   uint256 public swapTokensAtAmount = 4000 * (10**18);
810   uint256 public maxWalletToken = 220000 * (10**18);
811
812   address payable public buybackWallet =
payable(0xfebB0033c348ce91C19d7B075134ec7A7e044e1b);
813
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 810

low SEVERITY

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

Source File

- BB.sol

Locations

```
809   uint256 public swapTokensAtAmount = 4000 * (10**18);
810   uint256 public maxWalletToken = 220000 * (10**18);
811
812   address payable public buybackWallet =
payable(0xfebB0033c348ce91C19d7B075134ec7A7e044e1b);
813   address public deadWallet = 0x000000000000000000000000000000000000dEaD;
814
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 810

low SEVERITY

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

Source File

- BB.sol

Locations

```
809   uint256 public swapTokensAtAmount = 4000 * (10**18);
810   uint256 public maxWalletToken = 220000 * (10**18);
811
812   address payable public buybackWallet =
payable(0xfebB0033c348ce91C19d7B075134ec7A7e044e1b);
813   address public deadWallet = 0x000000000000000000000000000000000000dEaD;
814
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 861

low SEVERITY

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

Source File

- BB.sol

Locations

```
860   */
861   _createTotalSupply(owner(), 22000000 * (10**18));
862   }
863
864   function setLaunchStatus(bool launched_) public onlyOwner {
865
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 861

low SEVERITY

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

Source File

- BB.sol

Locations

```
860   */
861   _createTotalSupply(owner(), 22000000 * (10**18));
862   }
863
864   function setLaunchStatus(bool launched_) public onlyOwner {
865
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 888

low SEVERITY

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

Source File

- BB.sol

Locations

```
887   uint256 contractBalanceRecepient = balanceOf(to);
888   require(contractBalanceRecepient + amount <= maxWalletToken, "Exceeds maximum
wallet token amount.");
889   }
890
891   if(!_isExcludedFromFees[from] && !_isExcludedFromFees[to] && from==uniswapV2Pair){
892
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 915

low SEVERITY

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

Source File

- BB.sol

Locations

```
914   buyBackShare = amount.mul(devBuyFee).div(100);
915   buyBackTokens += buyBackShare;
916   super._transfer(from, address(this), buyBackShare);
917   }
918   if(liquidityBuyFee > 0) {
919
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 920

low SEVERITY

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

Source File

- BB.sol

Locations

```
919   liquidityShare = amount.mul(liquidityBuyFee).div(100);
920   liquidityTokens += liquidityShare;
921   super._transfer(from, address(this), liquidityShare);
922   }
923   }
924
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 927

low SEVERITY

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

Source File

- BB.sol

Locations

```
926   buyBackShare = amount.mul(devSellFee).div(100);
927   buyBackTokens += buyBackShare;
928   super._transfer(from, address(this), buyBackShare);
929   }
930   if(liquiditySellFee > 0) {
931
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 932

low SEVERITY

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

Source File

- BB.sol

Locations

```
931   liquidityShare = amount.mul(liquiditySellFee).div(100);
932   liquidityTokens += liquidityShare;
933   super._transfer(from, address(this), liquidityShare);
934   }
935   } else { //Transfer
936
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 938

low SEVERITY

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

Source File

- BB.sol

Locations

```
937   buyBackShare = amount.mul(devTransferFee).div(100);
938   buyBackTokens += buyBackShare;
939   super._transfer(from, address(this), buyBackShare);
940   }
941   if(liquidityTransferFee > 0) {
942
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 943

low SEVERITY

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

Source File

- BB.sol

Locations

```
942   liquidityShare = amount.mul(liquidityTransferFee).div(100);
943   liquidityTokens += liquidityShare;
944   super._transfer(from, address(this), liquidityShare);
945   }
946   }
947
```

SWC-101 | ARITHMETIC OPERATION "-=" DISCOVERED

LINE 973

low SEVERITY

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

Source File

- BB.sol

Locations

```
972   emit SwapAndLiquify(half, newBalance, otherHalf);
973   liquidityTokens -= swapTokensAtAmount;
974   }
975
976   if(buyBackTokens >= swapTokensAtAmount && contractTokenBalance >=
swapTokensAtAmount) {
977
```

SWC-101 | ARITHMETIC OPERATION "-=" DISCOVERED

LINE 978

low SEVERITY

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

Source File

- BB.sol

Locations

```
977   swapTokensForEth(swapTokensAtAmount, buybackWallet);
978   buyBackTokens -= swapTokensAtAmount;
979   }
980
981   }
982
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1001

low SEVERITY

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

Source File

- BB.sol

Locations

```
1000   address[] memory path = new address[](2);
1001   path[0] = address(this);
1002   path[1] = uniswapV2Router.WETH();
1003
1004   if(allowance(address(this), address(uniswapV2Router)) < tokenAmount) {
1005
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1002

low SEVERITY

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

Source File

- BB.sol

Locations

```
1001   path[0] = address(this);
1002   path[1] = uniswapV2Router.WETH();
1003
1004   if(allowance(address(this), address(uniswapV2Router)) < tokenAmount) {
1005   _approve(address(this), address(uniswapV2Router), ~uint256(0));
1006
```

DISCLAIMER

This report is subject to the terms and conditions (including without limitation, description of services, confidentiality, disclaimer and limitation of liability) set forth in the Services Agreement, or the scope of services, and terms and conditions provided to you ("Customer" or the "Company") in connection with the Agreement. This report provided in connection with the Services set forth in the Agreement shall be used by the Company only to the extent permitted under the terms and conditions set forth in the Agreement. This report may not be transmitted, disclosed, referred to, or relied upon by any person for any purposes, nor may copies be delivered to any other person other than the Company, without Sysfixed's prior written consent in each instance.

This report is not, nor should be considered, an "endorsement" or "disapproval" of any particular project or team. This report is not, nor should be considered, an indication of the economics or value of any "product" or "asset" created by any team or project that contracts Sysfixed to perform a security assessment. This report does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors, business, business model, or legal compliance.

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 us on the basis of what it says or doesn't say, or how we produced it, and it is important for you to conduct your own independent investigations before making any decisions. We go into more detail on this in the below disclaimer below – please make sure to read it in full.

This report should not be used in any way to make decisions around investment or involvement with any particular project. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

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 Sysfixed and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers, and other representatives) (Sysfixed) owe no duty of care.

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.