Blue Brilliant AI

Smart Contract Audit Report

SYSFIXED

25 Jan 2023

TABLE OF CONTENTS

AUDITED DETAILS

Audited Project

Project name	Token ticker	Blockchain
Blue Brilliant AI	BRILL	Binance Smart Chain

Addresses

Contract address	0x7b99409F607857F4dbf1980Ab2C272d5369E4ad5
Contract deployer address	0x5AE11a1B6787CFdC7905c3A23cdee3aA78C80d3F

Project Website

https://bluebrilliant.net/

Codebase

https://bscscan.com/address/0x7b99409F607857F4dbf1980Ab2C272d5369E4ad5#code

SUMMARY

Blue Brilliant AI is creating an innovative p2e platform with absolutely unique features. P2E, In-game NFT store, Staking, Crypto Casino. Working on a game with the integration of artificial intelligence! Buyback mechanism for price support! Buy/Sell tax: 6%! No Private Sale! 0% Team Tokens!

Contract Summary

Documentation Quality

Blue Brilliant AI 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 Blue Brilliant AI 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 195, 217, 242, 271, 272, 401, 402, 403, 404, 441, 472, 482, 493, 517, 528, 533, 546, 555, 563, 574, 581, 585, 605, 606, 608, 614, 615, 616, 623, 628, 633, 682, 692, 702, 734, 744, 753, 754 and 755.
- SWC-103 | Pragma statements can be allowed to float when a contract is intended on lines 8.
- SWC-110 | It is recommended to use of revert(), assert(), and require() in Solidity, and the new REVERT opcode in the EVM on lines 644, 645 and 745.
- SWC-120 | It is recommended to use external sources of randomness via oracles on lines 546 and 709.

CONCLUSION

We have audited the Blue Brilliant AI project which has released on January 2023 to discover issues and identify potential security vulnerabilities in Blue Brilliant AI Project. This process is used to find technical issues and security loopholes that find some common issues in the code.

The security audit report produced satisfactory results with low-risk issues.

The most common issue found in writing code on contracts that do not pose a big risk, writing on contracts is close to the standard of writing contracts in general. The low-level issues found are some arithmetic operation issues, a floating pragma is set, weak sources of randomness and out of bounds array access which the index access expression can cause an exception in case of 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.	ISSUE FOUND
Unchecked Call Return Value	SWC-104	The return value of a message call should be checked.	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
Assert Violation	SWC-110	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 Caller	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
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.	ISSUE FOUND
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

SMART CONTRACT ANALYSIS

Started	Tuesday Jan 24 2023 20:57:39 GMT+0000 (Coordinated Universal Time)
Finished	Wednesday Jan 25 2023 03:51:46 GMT+0000 (Coordinated Universal Time)
Mode	Standard
Main Source File	BlueBrilliantAI.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-103	A FLOATING PRAGMA IS SET.	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-120	POTENTIAL USE OF "BLOCK.NUMBER" AS SOURCE OF RANDOMNESS.	low	acknowledged
SWC-120	POTENTIAL USE OF "BLOCK.NUMBER" AS SOURCE OF RANDOMNESS.	low	acknowledged

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 195

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
194   require(currentAllowance >= amount, "BEP20: transfer amount exceeds allowance");
195   _approve(sender, _msgSender(), currentAllowance - amount);
196
197   return true;
198   }
199
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 217

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
216   {
217   _approve(_msgSender(), spender, _allowances[_msgSender()][spender] + addedValue);
218   return true;
219   }
220
221
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 242

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
241   require(currentAllowance >= subtractedValue, "BEP20: decreased allowance below
zero");
242   _approve(_msgSender(), spender, currentAllowance - subtractedValue);
243
244   return true;
245   }
246
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 271

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
270   require(senderBalance >= amount, "BEP20: transfer amount exceeds balance");
271   _balances[sender] = senderBalance - amount;
272   _balances[recipient] += amount;
273
274   emit Transfer(sender, recipient, amount);
275
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 272

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
271   _balances[sender] = senderBalance - amount;
272   _balances[recipient] += amount;
273
274   emit Transfer(sender, recipient, amount);
275   }
276
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 401

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
400
401   uint256 public tokenLiquidityThreshold = 1e5 * 10**18;
402   uint256 public maxBuyLimit = 1e8 * 10**18;
403   uint256 public maxSellLimit = 1e8 * 10**18;
404   uint256 public maxWalletLimit = 1e8 * 10**18;
405
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 402

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
401   uint256 public tokenLiquidityThreshold = 1e5 * 10**18;
402   uint256 public maxBuyLimit = 1e8 * 10**18;
403   uint256 public maxSellLimit = 1e8 * 10**18;
404   uint256 public maxWalletLimit = 1e8 * 10**18;
405
406
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 403

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
402   uint256 public maxBuyLimit = 1e8 * 10**18;
403   uint256 public maxSellLimit = 1e8 * 10**18;
404   uint256 public maxWalletLimit = 1e8 * 10**18;
405
406   uint256 public genesis_block;
407
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 404

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
403   uint256 public maxSellLimit = 1e8 * 10**18;
404   uint256 public maxWalletLimit = 1e8 * 10**18;
405
406   uint256 public genesis_block;
407   uint256 private deadline = 3;
408
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 441

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
440   constructor() BEP20("Blue Brilliant AI", "BRILL") {
441   _tokengeneration(msg.sender, 1e8 * 10**decimals());
442   exemptFee[msg.sender] = true;
443
444   IRouter _router = IRouter(0x10ED43C718714eb63d5aA57B78B54704E256024E);
445
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 472

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
471   require(currentAllowance >= amount, "BEP20: transfer amount exceeds allowance");
472   _approve(sender, _msgSender(), currentAllowance - amount);
473
474   return true;
475   }
476
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 482

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
481   {
482   _approve(_msgSender(), spender, _allowances[_msgSender()][spender] + addedValue);
483   return true;
484   }
485
486
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 493

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
492   require(currentAllowance >= subtractedValue, "BEP20: decreased allowance below
zero");
493   _approve(_msgSender(), spender, currentAllowance - subtractedValue);
494
495   return true;
496   }
497
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 517

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
516   require(
517   balanceOf(recipient) + amount <= maxWalletLimit,
518   "You are exceeding maxWalletLimit"
519   );
520   }
521
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 528

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
527   require(
528   balanceOf(recipient) + amount <= maxWalletLimit,
529   "You are exceeding maxWalletLimit"
530   );
531   }
532
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 533

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
532   if (coolDownEnabled) {
533   uint256 timePassed = block.timestamp - _lastSell[sender];
534   require(timePassed >= coolDownTime, "Cooldown enabled");
535   _lastSell[sender] = block.timestamp;
536   }
537
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 546

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
545   !exemptFee[recipient] &&
546   block.number < genesis_block + deadline;
547
548   //set fee to zero if fees in contract are handled or exempted
549   if (_interlock || exemptFee[sender] || exemptFee[recipient])
550
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 555

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
554   feeswap =
555   sellTaxes.liquidity +
556   sellTaxes.marketing +
557   sellTaxes.bb +
558   sellTaxes.dev;
559
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 563

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
562   feeswap =
563   taxes.liquidity +
564   taxes.marketing +
565   taxes.bb +
566   taxes.dev ;
567
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 574

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
573
574   fee = (amount * feesum) / 100;
575
576   //send fees if threshold has been reached
577   //don't do this on buys, breaks swap
578
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 581

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
580   //rest to recipient
581   super._transfer(sender, recipient, amount - fee);
582   if (fee > 0) {
583   //send the fee to the contract
584   if (feeswap > 0) {
585
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 585

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
584   if (feeswap > 0) {
585   uint256 feeAmount = (amount * feeswap) / 100;
586   super._transfer(sender, address(this), feeAmount);
587   }
588
589
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 605

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
604   // Split the contract balance into halves
605   uint256 denominator = feeswap * 2;
606   uint256 tokensToAddLiquidityWith = (contractBalance * swapTaxes.liquidity) /
607   denominator;
608   uint256 toSwap = contractBalance - tokensToAddLiquidityWith;
609
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 606

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
605   uint256 denominator = feeswap * 2;
606   uint256 tokensToAddLiquidityWith = (contractBalance * swapTaxes.liquidity) /
607   denominator;
608   uint256 toSwap = contractBalance - tokensToAddLiquidityWith;
609
610
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 608

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
607   denominator;
608   uint256 toSwap = contractBalance - tokensToAddLiquidityWith;
609
610   uint256 initialBalance = address(this).balance;
611
612
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 614

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
613
614   uint256 deltaBalance = address(this).balance - initialBalance;
615   uint256 unitBalance = deltaBalance / (denominator - swapTaxes.liquidity);
616   uint256 ethToAddLiquidityWith = unitBalance * swapTaxes.liquidity;
617
618
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 615

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
614   uint256 deltaBalance = address(this).balance - initialBalance;
615   uint256 unitBalance = deltaBalance / (denominator - swapTaxes.liquidity);
616   uint256 ethToAddLiquidityWith = unitBalance * swapTaxes.liquidity;
617
618   if (ethToAddLiquidityWith > 0) {
619
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 616

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
615   uint256 unitBalance = deltaBalance / (denominator - swapTaxes.liquidity);
616   uint256 ethToAddLiquidityWith = unitBalance * swapTaxes.liquidity;
617
618   if (ethToAddLiquidityWith > 0) {
619   // Add liquidity to pancake
620
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 623

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
622
623   uint256 marketingAmt = unitBalance * 2 * swapTaxes.marketing;
624   if (marketingAmt > 0) {
625   payable(marketingWallet).sendValue(marketingAmt);
626   }
627
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 628

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
627
628   uint256 bbAmt = unitBalance * 2 * swapTaxes.bb;
629   if (bbAmt > 0) {
630   payable(bbWallet).sendValue(bbAmt);
631   }
632
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 633

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
632
633   uint256 devAmt = unitBalance * 2 * swapTaxes.dev;
634   if (devAmt > 0) {
635   payable(devWallet).sendValue(devAmt);
636   }
637
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 682

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
681   require(new_amount <= 1e6, "Swap threshold amount should be lower or equal to 1% of
tokens");
682   tokenLiquidityThreshold = new_amount * 10**decimals();
683   }
684
685   function SetBuyTaxes(
686
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 692

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
691   taxes = Taxes(_marketing, _liquidity,  _bb, _dev);
692   require((_marketing + _liquidity + _bb + _dev) <= 10, "Must keep fees at 10% or
less");
693   }
694
695   function SetSellTaxes(
696
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 702

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
701   sellTaxes = Taxes(_marketing, _liquidity,  _bb,  _dev);
702   require((_marketing + _liquidity + _bb + _dev) <= 14, "Must keep fees at 14% or
less");
703   }
704
705   function EnableTrading() external onlyOwner {
706
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 734

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
733   function updateCooldown(bool state, uint256 time) external onlyOwner {
734   coolDownTime = time * 1 seconds;
735   coolDownEnabled = state;
736   require(time <= 300, "cooldown timer cannot exceed 5 minutes");
737   }
738
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 744

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
743   function bulkExemptFee(address[] memory accounts, bool state) external onlyOwner {
744   for (uint256 i = 0; i < accounts.length; i++) {
745   exemptFee[accounts[i]] = state;
746   }
747   }
748
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 753

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
752   require(maxWallet >= 1e6, "Cannot set max wallet amount lower than 1%");
753   maxBuyLimit = maxBuy * 10**decimals();
754   maxSellLimit = maxSell * 10**decimals();
755   maxWalletLimit = maxWallet * 10**decimals();
756   }
757
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 754

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
753   maxBuyLimit = maxBuy * 10**decimals();
754   maxSellLimit = maxSell * 10**decimals();
755   maxWalletLimit = maxWallet * 10**decimals();
756   }
757
758
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 755

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
754   maxSellLimit = maxSell * 10**decimals();
755   maxWalletLimit = maxWallet * 10**decimals();
756   }
757
758   function rescueBNB(uint256 weiAmount) external onlyOwner {
759
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 8

low SEVERITY

The current pragma Solidity directive is ""^0.8.8"". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

Source File

- BlueBrilliantAI.sol

Locations

```
7
8   pragma solidity ^0.8.8;
9
10   abstract contract Context {
11   function _msgSender() internal view virtual returns (address) {
12
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 644

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
643   address[] memory path = new address[](2);
644   path[0] = address(this);
645   path[1] = router.WETH();
646
647   _approve(address(this), address(router), tokenAmount);
648
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 645

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
644   path[0] = address(this);
645   path[1] = router.WETH();
646
647   _approve(address(this), address(router), tokenAmount);
648
649
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 745

low SEVERITY

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

Source File

- BlueBrilliantAI.sol

Locations

```
744   for (uint256 i = 0; i < accounts.length; i++) {
745   exemptFee[accounts[i]] = state;
746   }
747   }
748
749
```

SWC-120 | POTENTIAL USE OF "BLOCK.NUMBER" AS SOURCE OF RANDOMNESS.

LINE 546

low SEVERITY

The environment variable "block.number" looks like it might be used as a source of randomness. Note that the values of variables like coinbase, gaslimit, block number and timestamp are predictable and can be manipulated by a malicious miner. Also keep in mind that attackers know hashes of earlier blocks. Don't use any of those environment variables as sources of randomness and be aware that use of these variables introduces a certain level of trust into miners.

Source File

- BlueBrilliantAI.sol

Locations

```
545   !exemptFee[recipient] &&
546   block.number < genesis_block + deadline;
547
548   //set fee to zero if fees in contract are handled or exempted
549   if (_interlock || exemptFee[sender] || exemptFee[recipient])
```

SWC-120 | POTENTIAL USE OF "BLOCK.NUMBER" AS SOURCE OF RANDOMNESS.

LINE 709

low SEVERITY

The environment variable "block.number" looks like it might be used as a source of randomness. Note that the values of variables like coinbase, gaslimit, block number and timestamp are predictable and can be manipulated by a malicious miner. Also keep in mind that attackers know hashes of earlier blocks. Don't use any of those environment variables as sources of randomness and be aware that use of these variables introduces a certain level of trust into miners.

Source File

- BlueBrilliantAI.sol

Locations

```
708   providingLiquidity = true;
709   genesis_block = block.number;
710   }
711
712   function updatedeadline(uint256 _deadline) external onlyOwner {
```

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.