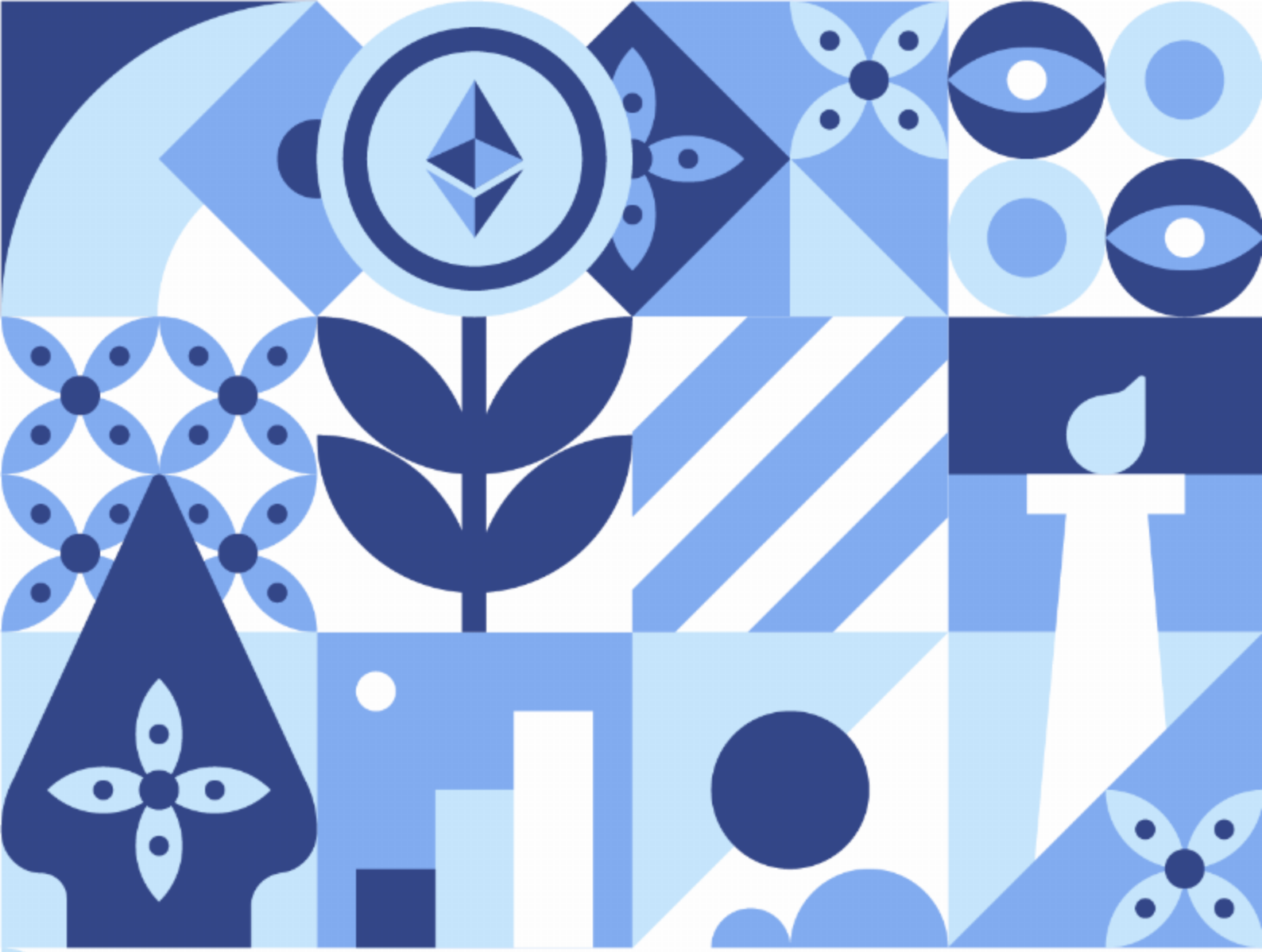

InfinityGaming

Smart Contract Audit Report

SYSFIXED

18 Dec 2021

TABLE OF CONTENTS

AUDITED DETAILS

Audited Project

Project name	Token ticker	Blockchain
InfinityGaming	PLAY	Ethereum

Addresses

Contract address	0x95b4e47025372ded4b73f9b5f0671b94a81445bc
Contract deployer address	0xA4E2A2733f2a538BF44F0AC4E326fE83F7a14C16

Project Website

http://infinitygaming.io/

Codebase

https://etherscan.io/address/0x95b4e47025372ded4b73f9b5f0671b94a81445bc#code

SUMMARY

Infinity Gaming is sponsoring a live tournament right now with a $7.5k prize pool. Less than 1 week after launch. 2k+ viewers watching real time and the streamer is pushing the PLAY token. 100k+ viewers expected to watch over 48 hours - giving it sustained exposure. Will bring insane awareness to the token, haven't seen this done before

Contract Summary

Documentation Quality

InfinityGaming 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 InfinityGaming with the discovery of several low issues.

Test Coverage

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

Audit Findings Summary

- SWC-100 SWC-108 | Explicitly define visibility for all state variables on lines 933 and 970.
- SWC-101 | It is recommended to use vetted safe math libraries for arithmetic operations consistently on lines 124, 160, 183, 184, 223, 263, 535, 923, 923, 923, 923, 924, 924, 973, 973, 973, 973, 974, 974, 974, 974, 975, 975, 975, 975, 1243, 1246, 1266, 1268, 1348, 1355, 1384, 1431, 1470, 1478, 1486, 1494, 1498, 1595, 1608, 1608, 1608, 1608, 1608, 1608, 1614, 1614, 1615, 1616, 1622, 1623, 1624, 1624, 1625, 1626, 1634, 1634, 1636, 1637, 1637, 1637, 1640, 1640, 1640, 1246 and 1268.
- SWC-103 | Pragma statements can be allowed to float when a contract is intended on lines 14.
- 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 1244, 1245, 1245, 1267, 1268, 1268, 1433, 1434, 1436, 1437, 1654 and 1655.
- SWC-115 | tx.origin should not be used for authorization, use msg.sender instead on lines 1555.

CONCLUSION

We have audited the InfinityGaming project released on December 2021 to discover issues and identify potential security vulnerabilities in InfinityGaming 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 InfinityGaming 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, a floating pragma is set, a state variable visibility is not set, 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 specifying 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 also avoiding 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.	ISSUE FOUND
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
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	Friday Dec 17 2021 16:16:03 GMT+0000 (Coordinated Universal Time)
Finished	Saturday Dec 18 2021 20:30:15 GMT+0000 (Coordinated Universal Time)
Mode	Standard
Main Source File	PLAY.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	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-101	COMPILER-REWRITABLE "<UINT> - 1" DISCOVERED	low	acknowledged
SWC-103	A FLOATING PRAGMA IS SET.	low	acknowledged
SWC-108	STATE VARIABLE VISIBILITY IS NOT SET.	low	acknowledged
SWC-108	STATE VARIABLE VISIBILITY IS NOT SET.	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-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 124

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
123   function add(uint256 a, uint256 b) internal pure returns (uint256) {
124   uint256 c = a + b;
125   require(c >= a, "SafeMath: addition overflow");
126
127   return c;
128
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 160

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
159   require(b <= a, errorMessage);
160   uint256 c = a - b;
161
162   return c;
163   }
164
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 183

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
182
183   uint256 c = a * b;
184   require(c / a == b, "SafeMath: multiplication overflow");
185
186   return c;
187
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 184

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
183   uint256 c = a * b;
184   require(c / a == b, "SafeMath: multiplication overflow");
185
186   return c;
187   }
188
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 223

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
222   require(b > 0, errorMessage);
223   uint256 c = a / b;
224   // assert(a == b * c + a % b); // There is no case in which this doesn't hold
225
226   return c;
227
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 263

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
262   require(b != 0, errorMessage);
263   return a % b;
264   }
265   }
266
267
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 535

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
534   _owner = address(0);
535   _lockTime = block.timestamp + time;
536   emit OwnershipTransferred(_owner, address(0));
537   }
538
539
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 923

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
922   uint256 private constant MAX = ~uint256(0);
923   uint256 private _tTotal = 1000000 * 10**6 * 10**9;
924   uint256 private _rTotal = (MAX - (MAX % _tTotal));
925   uint256 private _tFeeTotal;
926
927
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 923

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
922   uint256 private constant MAX = ~uint256(0);
923   uint256 private _tTotal = 1000000 * 10**6 * 10**9;
924   uint256 private _rTotal = (MAX - (MAX % _tTotal));
925   uint256 private _tFeeTotal;
926
927
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 923

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
922   uint256 private constant MAX = ~uint256(0);
923   uint256 private _tTotal = 1000000 * 10**6 * 10**9;
924   uint256 private _rTotal = (MAX - (MAX % _tTotal));
925   uint256 private _tFeeTotal;
926
927
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 923

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
922   uint256 private constant MAX = ~uint256(0);
923   uint256 private _tTotal = 1000000 * 10**6 * 10**9;
924   uint256 private _rTotal = (MAX - (MAX % _tTotal));
925   uint256 private _tFeeTotal;
926
927
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 924

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
923   uint256 private _tTotal = 1000000 * 10**6 * 10**9;
924   uint256 private _rTotal = (MAX - (MAX % _tTotal));
925   uint256 private _tFeeTotal;
926
927   address payable public _marketingAddress =
928
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 924

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
923   uint256 private _tTotal = 1000000 * 10**6 * 10**9;
924   uint256 private _rTotal = (MAX - (MAX % _tTotal));
925   uint256 private _tFeeTotal;
926
927   address payable public _marketingAddress =
928
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 973

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
972
973   uint256 public _maxTxAmount = 3000 * 10**6 * 10**9;
974   uint256 private numTokensSellToAddToLiquidity = 5000 * 10**6 * 10**9;
975   uint256 public _maxWalletSize = 15000 * 10**6 * 10**9;
976
977
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 973

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
972
973   uint256 public _maxTxAmount = 3000 * 10**6 * 10**9;
974   uint256 private numTokensSellToAddToLiquidity = 5000 * 10**6 * 10**9;
975   uint256 public _maxWalletSize = 15000 * 10**6 * 10**9;
976
977
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 973

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
972
973   uint256 public _maxTxAmount = 3000 * 10**6 * 10**9;
974   uint256 private numTokensSellToAddToLiquidity = 5000 * 10**6 * 10**9;
975   uint256 public _maxWalletSize = 15000 * 10**6 * 10**9;
976
977
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 973

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
972
973   uint256 public _maxTxAmount = 3000 * 10**6 * 10**9;
974   uint256 private numTokensSellToAddToLiquidity = 5000 * 10**6 * 10**9;
975   uint256 public _maxWalletSize = 15000 * 10**6 * 10**9;
976
977
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 974

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
973   uint256 public _maxTxAmount = 3000 * 10**6 * 10**9;
974   uint256 private numTokensSellToAddToLiquidity = 5000 * 10**6 * 10**9;
975   uint256 public _maxWalletSize = 15000 * 10**6 * 10**9;
976
977   event botAddedToBlacklist(address account);
978
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 974

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
973   uint256 public _maxTxAmount = 3000 * 10**6 * 10**9;
974   uint256 private numTokensSellToAddToLiquidity = 5000 * 10**6 * 10**9;
975   uint256 public _maxWalletSize = 15000 * 10**6 * 10**9;
976
977   event botAddedToBlacklist(address account);
978
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 974

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
973   uint256 public _maxTxAmount = 3000 * 10**6 * 10**9;
974   uint256 private numTokensSellToAddToLiquidity = 5000 * 10**6 * 10**9;
975   uint256 public _maxWalletSize = 15000 * 10**6 * 10**9;
976
977   event botAddedToBlacklist(address account);
978
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 974

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
973   uint256 public _maxTxAmount = 3000 * 10**6 * 10**9;
974   uint256 private numTokensSellToAddToLiquidity = 5000 * 10**6 * 10**9;
975   uint256 public _maxWalletSize = 15000 * 10**6 * 10**9;
976
977   event botAddedToBlacklist(address account);
978
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 975

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
974   uint256 private numTokensSellToAddToLiquidity = 5000 * 10**6 * 10**9;
975   uint256 public _maxWalletSize = 15000 * 10**6 * 10**9;
976
977   event botAddedToBlacklist(address account);
978   event botRemovedFromBlacklist(address account);
979
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 975

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
974   uint256 private numTokensSellToAddToLiquidity = 5000 * 10**6 * 10**9;
975   uint256 public _maxWalletSize = 15000 * 10**6 * 10**9;
976
977   event botAddedToBlacklist(address account);
978   event botRemovedFromBlacklist(address account);
979
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 975

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
974   uint256 private numTokensSellToAddToLiquidity = 5000 * 10**6 * 10**9;
975   uint256 public _maxWalletSize = 15000 * 10**6 * 10**9;
976
977   event botAddedToBlacklist(address account);
978   event botRemovedFromBlacklist(address account);
979
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 975

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
974   uint256 private numTokensSellToAddToLiquidity = 5000 * 10**6 * 10**9;
975   uint256 public _maxWalletSize = 15000 * 10**6 * 10**9;
976
977   event botAddedToBlacklist(address account);
978   event botRemovedFromBlacklist(address account);
979
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 1243

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1242   require(_isBlackListedBot[account], "Account is not blacklisted");
1243   for (uint256 i = 0; i < _blackListedBots.length; i++) {
1244   if (_blackListedBots[i] == account) {
1245   _blackListedBots[i] = _blackListedBots[
1246   _blackListedBots.length - 1
1247
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 1246

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1245   _blackListedBots[i] = _blackListedBots[
1246   _blackListedBots.length - 1
1247   ];
1248   _isBlackListedBot[account] = false;
1249   _blackListedBots.pop();
1250
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 1266

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1265   require(_isExcluded[account], "Account is not excluded");
1266   for (uint256 i = 0; i < _excluded.length; i++) {
1267   if (_excluded[i] == account) {
1268   _excluded[i] = _excluded[_excluded.length - 1];
1269   _tOwned[account] = 0;
1270
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 1268

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1267   if (_excluded[i] == account) {
1268   _excluded[i] = _excluded[_excluded.length - 1];
1269   _tOwned[account] = 0;
1270   _isExcluded[account] = false;
1271   _excluded.pop();
1272
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1348

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1347   function setMaxTxPercent(uint256 maxTxPercent) external onlyOwner {
1348   _maxTxAmount = _tTotal.mul(maxTxPercent).div(10**3);
1349   }
1350
1351   function _setMaxWalletSizePercent(uint256 maxWalletSize)
1352
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1355

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1354   {
1355   _maxWalletSize = _tTotal.mul(maxWalletSize).div(10**3);
1356   }
1357
1358   function setSwapAndLiquifyEnabled(bool _enabled) public onlyOwner {
1359
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1384

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1383   uint256 tLiquidity = calculateLiquidityFee(tAmount);
1384   uint256 tWallet = calculateMarketingFee(tAmount) +
1385   calculateDevFee(tAmount);
1386   uint256 tDonation = calculateDonationFee(tAmount);
1387   uint256 tTransferAmount = tAmount.sub(tFee).sub(tLiquidity);
1388
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 1431

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1430   uint256 tSupply = _tTotal;
1431   for (uint256 i = 0; i < _excluded.length; i++) {
1432   if (
1433   _rOwned[_excluded[i]] > rSupply ||
1434   _tOwned[_excluded[i]] > tSupply
1435
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1470

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1469   function calculateTaxFee(uint256 _amount) private view returns (uint256) {
1470   return _amount.mul(_taxFee).div(10**2);
1471   }
1472
1473   function calculateLiquidityFee(uint256 _amount)
1474
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1478

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1477   {
1478   return _amount.mul(_liquidityFee).div(10**2);
1479   }
1480
1481   function calculateMarketingFee(uint256 _amount)
1482
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1486

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1485   {
1486   return _amount.mul(_marketingFee).div(10**2);
1487   }
1488
1489   function calculateDonationFee(uint256 _amount)
1490
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1494

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1493   {
1494   return _amount.mul(_donationFee).div(10**2);
1495   }
1496
1497   function calculateDevFee(uint256 _amount) private view returns (uint256) {
1498
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1498

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1497   function calculateDevFee(uint256 _amount) private view returns (uint256) {
1498   return _amount.mul(_devFee).div(10**2);
1499   }
1500
1501   function removeAllFee() private {
1502
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1595

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1594   require(
1595   amount + balanceOf(to) <= _maxWalletSize,
1596   "Recipient exceeds max wallet size."
1597   );
1598   }
1599
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1608

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1607   // Split the contract balance into halves
1608   uint256 denominator = (buyFee.liquidity +
1609   sellFee.liquidity +
1610   buyFee.marketing +
1611   sellFee.marketing +
1612
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1608

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1607   // Split the contract balance into halves
1608   uint256 denominator = (buyFee.liquidity +
1609   sellFee.liquidity +
1610   buyFee.marketing +
1611   sellFee.marketing +
1612
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1608

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1607   // Split the contract balance into halves
1608   uint256 denominator = (buyFee.liquidity +
1609   sellFee.liquidity +
1610   buyFee.marketing +
1611   sellFee.marketing +
1612
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1608

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1607   // Split the contract balance into halves
1608   uint256 denominator = (buyFee.liquidity +
1609   sellFee.liquidity +
1610   buyFee.marketing +
1611   sellFee.marketing +
1612
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1608

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1607   // Split the contract balance into halves
1608   uint256 denominator = (buyFee.liquidity +
1609   sellFee.liquidity +
1610   buyFee.marketing +
1611   sellFee.marketing +
1612
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1608

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1607   // Split the contract balance into halves
1608   uint256 denominator = (buyFee.liquidity +
1609   sellFee.liquidity +
1610   buyFee.marketing +
1611   sellFee.marketing +
1612
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1614

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1613   sellFee.dev) * 2;
1614   uint256 tokensToAddLiquidityWith = (tokens *
1615   (buyFee.liquidity + sellFee.liquidity)) / denominator;
1616   uint256 toSwap = tokens - tokensToAddLiquidityWith;
1617
1618
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1614

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1613   sellFee.dev) * 2;
1614   uint256 tokensToAddLiquidityWith = (tokens *
1615   (buyFee.liquidity + sellFee.liquidity)) / denominator;
1616   uint256 toSwap = tokens - tokensToAddLiquidityWith;
1617
1618
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1615

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1614   uint256 tokensToAddLiquidityWith = (tokens *
1615   (buyFee.liquidity + sellFee.liquidity)) / denominator;
1616   uint256 toSwap = tokens - tokensToAddLiquidityWith;
1617
1618   uint256 initialBalance = address(this).balance;
1619
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 1616

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1615   (buyFee.liquidity + sellFee.liquidity)) / denominator;
1616   uint256 toSwap = tokens - tokensToAddLiquidityWith;
1617
1618   uint256 initialBalance = address(this).balance;
1619
1620
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 1622

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1621
1622   uint256 deltaBalance = address(this).balance - initialBalance;
1623   uint256 unitBalance = deltaBalance /
1624   (denominator - (buyFee.liquidity + sellFee.liquidity));
1625   uint256 bnbToAddLiquidityWith = unitBalance *
1626
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1623

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1622   uint256 deltaBalance = address(this).balance - initialBalance;
1623   uint256 unitBalance = deltaBalance /
1624   (denominator - (buyFee.liquidity + sellFee.liquidity));
1625   uint256 bnbToAddLiquidityWith = unitBalance *
1626   (buyFee.liquidity + sellFee.liquidity);
1627
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 1624

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1623   uint256 unitBalance = deltaBalance /
1624   (denominator - (buyFee.liquidity + sellFee.liquidity));
1625   uint256 bnbToAddLiquidityWith = unitBalance *
1626   (buyFee.liquidity + sellFee.liquidity);
1627
1628
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1624

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1623   uint256 unitBalance = deltaBalance /
1624   (denominator - (buyFee.liquidity + sellFee.liquidity));
1625   uint256 bnbToAddLiquidityWith = unitBalance *
1626   (buyFee.liquidity + sellFee.liquidity);
1627
1628
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1625

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1624   (denominator - (buyFee.liquidity + sellFee.liquidity));
1625   uint256 bnbToAddLiquidityWith = unitBalance *
1626   (buyFee.liquidity + sellFee.liquidity);
1627
1628   if (bnbToAddLiquidityWith > 0) {
1629
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1626

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1625   uint256 bnbToAddLiquidityWith = unitBalance *
1626   (buyFee.liquidity + sellFee.liquidity);
1627
1628   if (bnbToAddLiquidityWith > 0) {
1629   // Add liquidity to pancake
1630
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1634

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1633   // Send ETH to marketing
1634   uint256 marketingAmt = unitBalance *
1635   2 *
1636   (buyFee.marketing + sellFee.marketing);
1637   uint256 devAmt = unitBalance * 2 * (buyFee.dev + sellFee.dev) >
1638
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1634

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1633   // Send ETH to marketing
1634   uint256 marketingAmt = unitBalance *
1635   2 *
1636   (buyFee.marketing + sellFee.marketing);
1637   uint256 devAmt = unitBalance * 2 * (buyFee.dev + sellFee.dev) >
1638
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1636

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1635   2 *
1636   (buyFee.marketing + sellFee.marketing);
1637   uint256 devAmt = unitBalance * 2 * (buyFee.dev + sellFee.dev) >
1638   address(this).balance
1639   ? address(this).balance
1640
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1637

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1636   (buyFee.marketing + sellFee.marketing);
1637   uint256 devAmt = unitBalance * 2 * (buyFee.dev + sellFee.dev) >
1638   address(this).balance
1639   ? address(this).balance
1640   : unitBalance * 2 * (buyFee.dev + sellFee.dev);
1641
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1637

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1636   (buyFee.marketing + sellFee.marketing);
1637   uint256 devAmt = unitBalance * 2 * (buyFee.dev + sellFee.dev) >
1638   address(this).balance
1639   ? address(this).balance
1640   : unitBalance * 2 * (buyFee.dev + sellFee.dev);
1641
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1637

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1636   (buyFee.marketing + sellFee.marketing);
1637   uint256 devAmt = unitBalance * 2 * (buyFee.dev + sellFee.dev) >
1638   address(this).balance
1639   ? address(this).balance
1640   : unitBalance * 2 * (buyFee.dev + sellFee.dev);
1641
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1640

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1639   ? address(this).balance
1640   : unitBalance * 2 * (buyFee.dev + sellFee.dev);
1641
1642   if (marketingAmt > 0) {
1643   payable(_marketingAddress).transfer(marketingAmt);
1644
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1640

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1639   ? address(this).balance
1640   : unitBalance * 2 * (buyFee.dev + sellFee.dev);
1641
1642   if (marketingAmt > 0) {
1643   payable(_marketingAddress).transfer(marketingAmt);
1644
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1640

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1639   ? address(this).balance
1640   : unitBalance * 2 * (buyFee.dev + sellFee.dev);
1641
1642   if (marketingAmt > 0) {
1643   payable(_marketingAddress).transfer(marketingAmt);
1644
```

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

LINE 1246

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1245   _blackListedBots[i] = _blackListedBots[
1246   _blackListedBots.length - 1
1247   ];
1248   _isBlackListedBot[account] = false;
1249   _blackListedBots.pop();
1250
```

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

LINE 1268

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1267   if (_excluded[i] == account) {
1268   _excluded[i] = _excluded[_excluded.length - 1];
1269   _tOwned[account] = 0;
1270   _isExcluded[account] = false;
1271   _excluded.pop();
1272
```

SWC-103 | A FLOATING PRAGMA IS SET.

LINE 14

low SEVERITY

The current pragma Solidity directive is ""^0.8.10"". 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

- PLAY.sol

Locations

```
13
14   pragma solidity ^0.8.10;
15
16   // SPDX-License-Identifier: Unlicensed
17   interface IERC20 {
18
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 933

low SEVERITY

It is best practice to set the visibility of state variables explicitly. The default visibility for "_partnershipswallet" is internal. Other possible visibility settings are public and private.

Source File

- PLAY.sol

Locations

```
932   payable(address(0x9448Bc5e6D09270d8E31e24991B3941Ea88d9604));
933   address _partnershipswallet =
934   payable(address(0x3fcaC6ecC2d0365EDD177a3F34a460d9266E6C50));
935   address private _donationAddress =
936   0x000000000000000000000000000000000000dEaD;
937
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 970

low SEVERITY

It is best practice to set the visibility of state variables explicitly. The default visibility for "inSwapAndLiquify" is internal. Other possible visibility settings are public and private.

Source File

- PLAY.sol

Locations

```
969
970   bool inSwapAndLiquify;
971   bool public swapAndLiquifyEnabled = true;
972
973   uint256 public _maxTxAmount = 3000 * 10**6 * 10**9;
974
```

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

LINE 1555

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

- PLAY.sol

Locations

```
1554   require(!_isBlackListedBot[msg.sender], "blacklisted");
1555   require(!_isBlackListedBot[tx.origin], "blacklisted");
1556
1557   // is the token balance of this contract address over the min number of
1558   // tokens that we need to initiate a swap + liquidity lock?
1559
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1244

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1243   for (uint256 i = 0; i < _blackListedBots.length; i++) {
1244   if (_blackListedBots[i] == account) {
1245   _blackListedBots[i] = _blackListedBots[
1246   _blackListedBots.length - 1
1247   ];
1248
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1245

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1244   if (_blackListedBots[i] == account) {
1245   _blackListedBots[i] = _blackListedBots[
1246   _blackListedBots.length - 1
1247   ];
1248   _isBlackListedBot[account] = false;
1249
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1245

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1244   if (_blackListedBots[i] == account) {
1245   _blackListedBots[i] = _blackListedBots[
1246   _blackListedBots.length - 1
1247   ];
1248   _isBlackListedBot[account] = false;
1249
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1267

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1266   for (uint256 i = 0; i < _excluded.length; i++) {
1267   if (_excluded[i] == account) {
1268   _excluded[i] = _excluded[_excluded.length - 1];
1269   _tOwned[account] = 0;
1270   _isExcluded[account] = false;
1271
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1268

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1267   if (_excluded[i] == account) {
1268   _excluded[i] = _excluded[_excluded.length - 1];
1269   _tOwned[account] = 0;
1270   _isExcluded[account] = false;
1271   _excluded.pop();
1272
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1268

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1267   if (_excluded[i] == account) {
1268   _excluded[i] = _excluded[_excluded.length - 1];
1269   _tOwned[account] = 0;
1270   _isExcluded[account] = false;
1271   _excluded.pop();
1272
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1433

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1432   if (
1433   _rOwned[_excluded[i]] > rSupply ||
1434   _tOwned[_excluded[i]] > tSupply
1435   ) return (_rTotal, _tTotal);
1436   rSupply = rSupply.sub(_rOwned[_excluded[i]]);
1437
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1434

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1433   _rOwned[_excluded[i]] > rSupply ||
1434   _tOwned[_excluded[i]] > tSupply
1435   ) return (_rTotal, _tTotal);
1436   rSupply = rSupply.sub(_rOwned[_excluded[i]]);
1437   tSupply = tSupply.sub(_tOwned[_excluded[i]]);
1438
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1436

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1435   ) return (_rTotal, _tTotal);
1436   rSupply = rSupply.sub(_rOwned[_excluded[i]]);
1437   tSupply = tSupply.sub(_tOwned[_excluded[i]]);
1438   }
1439   if (rSupply < _rTotal.div(_tTotal)) return (_rTotal, _tTotal);
1440
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1437

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1436   rSupply = rSupply.sub(_rOwned[_excluded[i]]);
1437   tSupply = tSupply.sub(_tOwned[_excluded[i]]);
1438   }
1439   if (rSupply < _rTotal.div(_tTotal)) return (_rTotal, _tTotal);
1440   return (rSupply, tSupply);
1441
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1654

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1653   address[] memory path = new address[](2);
1654   path[0] = address(this);
1655   path[1] = uniswapV2Router.WETH();
1656
1657   _approve(address(this), address(uniswapV2Router), tokenAmount);
1658
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1655

low SEVERITY

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

Source File

- PLAY.sol

Locations

```
1654   path[0] = address(this);
1655   path[1] = uniswapV2Router.WETH();
1656
1657   _approve(address(this), address(uniswapV2Router), tokenAmount);
1658
1659
```

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.