



# Ethereum Shillings Smart Contract Audit Report

# 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
Ethereum Shillings	eSHILL	Ethereum

## Addresses

Contract address	0xbe3ee40726c1578581a92b4a9fa4acfce65a4e10
Contract deployer address	0xA7eb8C15B45E8c806aC4810788B695aEFEA0F425

## Project Website

<https://ethereumshillings.com/>

## Codebase

<https://etherscan.io/address/0xbe3ee40726c1578581a92b4a9fa4acfce65a4e10#code>

# SUMMARY

Ethereum Shillings, eSHILL, is an innovative, disruptive, community-driven, decentralized finance (Defi) token built on the Ethereum network. Our mission is to improve financial inclusion through the globalization of simple peer-to-peer transactions in the cryptocurrency space.

## Contract Summary

### Documentation Quality

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

- The technical description is provided clearly and structured and also don't have any high risk issue.

### Code Quality

The Overall quality of the basecode is standard.

- Standard solidity basecode and rules are already followed by Ethereum Shillings 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 744.
- SWC-101 | It is recommended to use vetted safe math libraries for arithmetic operations consistently on lines 105, 137, 160, 161, 196, 232, 459, 701, 701, 701, 701, 702, 702, 734, 734, 734, 734, 735, 735, 735, 736, 736, 736, 736, 737, 737, 737, 737, 738, 738, 738, 738, 739, 739, 739, 739, 747, 747, 747, 747, 748, 748, 748, 748, 781, 928, 930, 932, 933, 1007, 1013, 1019 and 1095.
- SWC-103 | Pragma statements can be allowed to float when a contract is intended on lines 6.
- 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 929, 930, 931, 933, 933, 954, 954, 954, 955, 955, 955, 955, 955, 955, 1143 and 1144.

## CONCLUSION

We have audited the Ethereum Shillings project released on December 2021 to discover issues and identify potential security vulnerabilities in Ethereum Shillings 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 Ethereum Shillings 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 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.	<b>ISSUE FOUND</b>
Integer Overflow and Underflow	SWC-101	If unchecked math is used, all math operations should be safe from overflows and underflows.	<b>ISSUE FOUND</b>
Outdated Compiler Version	SWC-102	It is recommended to use a recent version of the Solidity compiler.	<b>PASS</b>
Floating Pragma	SWC-103	Contracts should be deployed with the same compiler version and flags that they have been tested thoroughly.	<b>ISSUE FOUND</b>
Unchecked Call Return Value	SWC-104	The return value of a message call should be checked.	<b>PASS</b>
Unprotected Ether Withdrawal	SWC-105	Due to missing or insufficient access controls, malicious parties can withdraw from the contract.	<b>PASS</b>
SELFDESTRUCT Instruction	SWC-106	The contract should not be self-destructible while it has funds belonging to users.	<b>PASS</b>
Reentrancy	SWC-107	Check effect interaction pattern should be followed if the code performs recursive call.	<b>PASS</b>
Uninitialized Storage Pointer	SWC-109	Uninitialized local storage variables can point to unexpected storage locations in the contract.	<b>PASS</b>
Assert Violation	SWC-110 SWC-123	Properly functioning code should never reach a failing assert statement.	<b>ISSUE FOUND</b>
Deprecated Solidity Functions	SWC-111	Deprecated built-in functions should never be used.	<b>PASS</b>
Delegate call to Untrusted Callee	SWC-112	Delegatecalls should only be allowed to trusted addresses.	<b>PASS</b>

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 <code>abi.encodePacked()</code> with multiple variable length arguments can, in certain situations, lead to a hash collision.	PASS
Hardcoded gas amount	SWC-134	The <code>transfer()</code> and <code>send()</code> 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	Saturday Dec 11 2021 09:45:34 GMT+0000 (Coordinated Universal Time)
Finished	Sunday Dec 12 2021 10:39:57 GMT+0000 (Coordinated Universal Time)
Mode	Standard
Main Source File	eSHILLTOKEN.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

LINE 105

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
104 function add(uint256 a, uint256 b) internal pure returns (uint256) {
105     uint256 c = a + b;
106     require(c >= a, "SafeMath: addition overflow");
107
108     return c;
109 }
```

# SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 137

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
136   require(b <= a, errorMessage);
137   uint256 c = a - b;
138
139   return c;
140   }
141
```

## SWC-101 | ARITHMETIC OPERATION "\*" DISCOVERED

LINE 160

### low SEVERITY

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

### Source File

- eSHILLTOKEN.sol

### Locations

```
159
160  uint256 c = a * b;
161  require(c / a == b, "SafeMath: multiplication overflow");
162
163  return c;
164
```

# SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 161

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
160  uint256 c = a * b;
161  require(c / a == b, "SafeMath: multiplication overflow");
162
163  return c;
164  }
165
```



# SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 196

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
195   require(b > 0, errorMessage);
196   uint256 c = a / b;
197   // assert(a == b * c + a % b); // There is no case in which this doesn't hold
198
199   return c;
200
```

# SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 232

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
231     require(b != 0, errorMessage);
232     return a % b;
233 }
234 }
235
236
```

# SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 459

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
458  _owner = address(0);
459  _lockTime = now + time;
460  emit OwnershipTransferred(_owner, address(0));
461  }
462
463
```

# SWC-101 | ARITHMETIC OPERATION "\*" DISCOVERED

LINE 701

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
700 uint256 private constant MAX = ~uint256(0);
701 uint256 private _tTotal = 100 * 10**15 * 10**9;
702 uint256 private _rTotal = (MAX - (MAX % _tTotal));
703 uint256 private _tFeeTotal;
704
705
```

# SWC-101 | ARITHMETIC OPERATION "\*" DISCOVERED

LINE 701

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
700 uint256 private constant MAX = ~uint256(0);
701 uint256 private _tTotal = 100 * 10**15 * 10**9;
702 uint256 private _rTotal = (MAX - (MAX % _tTotal));
703 uint256 private _tFeeTotal;
704
705
```

# SWC-101 | ARITHMETIC OPERATION "\*\*" DISCOVERED

LINE 701

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
700 uint256 private constant MAX = ~uint256(0);
701 uint256 private _tTotal = 100 * 10**15 * 10**9;
702 uint256 private _rTotal = (MAX - (MAX % _tTotal));
703 uint256 private _tFeeTotal;
704
705
```

# SWC-101 | ARITHMETIC OPERATION "\*\*" DISCOVERED

LINE 701

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
700 uint256 private constant MAX = ~uint256(0);
701 uint256 private _tTotal = 100 * 10**15 * 10**9;
702 uint256 private _rTotal = (MAX - (MAX % _tTotal));
703 uint256 private _tFeeTotal;
704
705
```

# SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 702

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
701  uint256 private _tTotal = 100 * 10**15 * 10**9;  
702  uint256 private _rTotal = (MAX - (MAX % _tTotal));  
703  uint256 private _tFeeTotal;  
704  
705  string private _name = "Ethereum Shillings";  
706
```



# SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 702

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
701  uint256 private _tTotal = 100 * 10**15 * 10**9;
702  uint256 private _rTotal = (MAX - (MAX % _tTotal));
703  uint256 private _tFeeTotal;
704
705  string private _name = "Ethereum Shillings";
706
```

# SWC-101 | ARITHMETIC OPERATION "\*" DISCOVERED

LINE 734

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
733
734 uint256 private constant _devAllocation = 3 * 10**15 * 10**9;
735 uint256 private constant _listingsAllocation = 5 * 10**15 * 10**9;
736 uint256 private constant _marketingAllocation = 3 * 10**15 * 10**9;
737 uint256 private constant _privateSaleAllocation = 29 * 10**15 * 10**9;
738
```

# SWC-101 | ARITHMETIC OPERATION "\*" DISCOVERED

LINE 734

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
733
734 uint256 private constant _devAllocation = 3 * 10**15 * 10**9;
735 uint256 private constant _listingsAllocation = 5 * 10**15 * 10**9;
736 uint256 private constant _marketingAllocation = 3 * 10**15 * 10**9;
737 uint256 private constant _privateSaleAllocation = 29 * 10**15 * 10**9;
738
```

# SWC-101 | ARITHMETIC OPERATION "\*\*" DISCOVERED

LINE 734

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
733
734 uint256 private constant _devAllocation = 3 * 10**15 * 10**9;
735 uint256 private constant _listingsAllocation = 5 * 10**15 * 10**9;
736 uint256 private constant _marketingAllocation = 3 * 10**15 * 10**9;
737 uint256 private constant _privateSaleAllocation = 29 * 10**15 * 10**9;
738
```

# SWC-101 | ARITHMETIC OPERATION "\*\*" DISCOVERED

LINE 734

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
733
734 uint256 private constant _devAllocation = 3 * 10**15 * 10**9;
735 uint256 private constant _listingsAllocation = 5 * 10**15 * 10**9;
736 uint256 private constant _marketingAllocation = 3 * 10**15 * 10**9;
737 uint256 private constant _privateSaleAllocation = 29 * 10**15 * 10**9;
738
```

# SWC-101 | ARITHMETIC OPERATION "\*" DISCOVERED

LINE 735

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
734 uint256 private constant _devAllocation = 3 * 10**15 * 10**9;  
735 uint256 private constant _listingsAllocation = 5 * 10**15 * 10**9;  
736 uint256 private constant _marketingAllocation = 3 * 10**15 * 10**9;  
737 uint256 private constant _privateSaleAllocation = 29 * 10**15 * 10**9;  
738 uint256 private constant _LPAllocation = 25 * 10**15 * 10**9;  
739
```

# SWC-101 | ARITHMETIC OPERATION "\*" DISCOVERED

LINE 735

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
734 uint256 private constant _devAllocation = 3 * 10**15 * 10**9;
735 uint256 private constant _listingsAllocation = 5 * 10**15 * 10**9;
736 uint256 private constant _marketingAllocation = 3 * 10**15 * 10**9;
737 uint256 private constant _privateSaleAllocation = 29 * 10**15 * 10**9;
738 uint256 private constant _LPAllocation = 25 * 10**15 * 10**9;
739
```

# SWC-101 | ARITHMETIC OPERATION "\*\*" DISCOVERED

LINE 735

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
734 uint256 private constant _devAllocation = 3 * 10**15 * 10**9;
735 uint256 private constant _listingsAllocation = 5 * 10**15 * 10**9;
736 uint256 private constant _marketingAllocation = 3 * 10**15 * 10**9;
737 uint256 private constant _privateSaleAllocation = 29 * 10**15 * 10**9;
738 uint256 private constant _LPAllocation = 25 * 10**15 * 10**9;
739
```



# SWC-101 | ARITHMETIC OPERATION "\*\*" DISCOVERED

LINE 735

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
734 uint256 private constant _devAllocation = 3 * 10**15 * 10**9;  
735 uint256 private constant _listingsAllocation = 5 * 10**15 * 10**9;  
736 uint256 private constant _marketingAllocation = 3 * 10**15 * 10**9;  
737 uint256 private constant _privateSaleAllocation = 29 * 10**15 * 10**9;  
738 uint256 private constant _LPAllocation = 25 * 10**15 * 10**9;  
739
```

# SWC-101 | ARITHMETIC OPERATION "\*" DISCOVERED

LINE 736

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
735 uint256 private constant _listingsAllocation = 5 * 10**15 * 10**9;  
736 uint256 private constant _marketingAllocation = 3 * 10**15 * 10**9;  
737 uint256 private constant _privateSaleAllocation = 29 * 10**15 * 10**9;  
738 uint256 private constant _LPAllocation = 25 * 10**15 * 10**9;  
739 uint256 private constant _burnAllocation = 35 * 10**15 * 10**9;  
740
```

# SWC-101 | ARITHMETIC OPERATION "\*" DISCOVERED

LINE 736

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
735 uint256 private constant _listingsAllocation = 5 * 10**15 * 10**9;  
736 uint256 private constant _marketingAllocation = 3 * 10**15 * 10**9;  
737 uint256 private constant _privateSaleAllocation = 29 * 10**15 * 10**9;  
738 uint256 private constant _LPAllocation = 25 * 10**15 * 10**9;  
739 uint256 private constant _burnAllocation = 35 * 10**15 * 10**9;  
740
```

# SWC-101 | ARITHMETIC OPERATION "\*\*" DISCOVERED

LINE 736

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
735 uint256 private constant _listingsAllocation = 5 * 10**15 * 10**9;  
736 uint256 private constant _marketingAllocation = 3 * 10**15 * 10**9;  
737 uint256 private constant _privateSaleAllocation = 29 * 10**15 * 10**9;  
738 uint256 private constant _LPAllocation = 25 * 10**15 * 10**9;  
739 uint256 private constant _burnAllocation = 35 * 10**15 * 10**9;  
740
```

# SWC-101 | ARITHMETIC OPERATION "\*\*" DISCOVERED

LINE 736

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
735 uint256 private constant _listingsAllocation = 5 * 10**15 * 10**9;  
736 uint256 private constant _marketingAllocation = 3 * 10**15 * 10**9;  
737 uint256 private constant _privateSaleAllocation = 29 * 10**15 * 10**9;  
738 uint256 private constant _LPAllocation = 25 * 10**15 * 10**9;  
739 uint256 private constant _burnAllocation = 35 * 10**15 * 10**9;  
740
```

# SWC-101 | ARITHMETIC OPERATION "\*" DISCOVERED

LINE 737

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
736 uint256 private constant _marketingAllocation = 3 * 10**15 * 10**9;  
737 uint256 private constant _privateSaleAllocation = 29 * 10**15 * 10**9;  
738 uint256 private constant _LPAllocation = 25 * 10**15 * 10**9;  
739 uint256 private constant _burnAllocation = 35 * 10**15 * 10**9;  
740  
741
```

# SWC-101 | ARITHMETIC OPERATION "\*" DISCOVERED

LINE 737

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
736 uint256 private constant _marketingAllocation = 3 * 10**15 * 10**9;  
737 uint256 private constant _privateSaleAllocation = 29 * 10**15 * 10**9;  
738 uint256 private constant _LPAllocation = 25 * 10**15 * 10**9;  
739 uint256 private constant _burnAllocation = 35 * 10**15 * 10**9;  
740  
741
```

# SWC-101 | ARITHMETIC OPERATION "\*\*" DISCOVERED

LINE 737

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
736 uint256 private constant _marketingAllocation = 3 * 10**15 * 10**9;  
737 uint256 private constant _privateSaleAllocation = 29 * 10**15 * 10**9;  
738 uint256 private constant _LPAllocation = 25 * 10**15 * 10**9;  
739 uint256 private constant _burnAllocation = 35 * 10**15 * 10**9;  
740  
741
```



# SWC-101 | ARITHMETIC OPERATION "\*\*" DISCOVERED

LINE 737

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
736 uint256 private constant _marketingAllocation = 3 * 10**15 * 10**9;  
737 uint256 private constant _privateSaleAllocation = 29 * 10**15 * 10**9;  
738 uint256 private constant _LPAllocation = 25 * 10**15 * 10**9;  
739 uint256 private constant _burnAllocation = 35 * 10**15 * 10**9;  
740  
741
```

# SWC-101 | ARITHMETIC OPERATION "\*" DISCOVERED

LINE 738

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
737 uint256 private constant _privateSaleAllocation = 29 * 10**15 * 10**9;  
738 uint256 private constant _LPAllocation = 25 * 10**15 * 10**9;  
739 uint256 private constant _burnAllocation = 35 * 10**15 * 10**9;  
740  
741 IUniswapV2Router02 public immutable uniswapV2Router;  
742
```

# SWC-101 | ARITHMETIC OPERATION "\*" DISCOVERED

LINE 738

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
737 uint256 private constant _privateSaleAllocation = 29 * 10**15 * 10**9;  
738 uint256 private constant _LPAllocation = 25 * 10**15 * 10**9;  
739 uint256 private constant _burnAllocation = 35 * 10**15 * 10**9;  
740  
741 IUniswapV2Router02 public immutable uniswapV2Router;  
742
```

# SWC-101 | ARITHMETIC OPERATION "\*\*" DISCOVERED

LINE 738

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
737 uint256 private constant _privateSaleAllocation = 29 * 10**15 * 10**9;  
738 uint256 private constant _LPAllocation = 25 * 10**15 * 10**9;  
739 uint256 private constant _burnAllocation = 35 * 10**15 * 10**9;  
740  
741 IUniswapV2Router02 public immutable uniswapV2Router;  
742
```

# SWC-101 | ARITHMETIC OPERATION "\*\*" DISCOVERED

LINE 738

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
737 uint256 private constant _privateSaleAllocation = 29 * 10**15 * 10**9;  
738 uint256 private constant _LPAllocation = 25 * 10**15 * 10**9;  
739 uint256 private constant _burnAllocation = 35 * 10**15 * 10**9;  
740  
741 IUniswapV2Router02 public immutable uniswapV2Router;  
742
```

# SWC-101 | ARITHMETIC OPERATION "\*" DISCOVERED

LINE 739

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
738 uint256 private constant _LPAllocation = 25 * 10**15 * 10**9;
739 uint256 private constant _burnAllocation = 35 * 10**15 * 10**9;
740
741 IUniswapV2Router02 public immutable uniswapV2Router;
742 address public immutable uniswapV2Pair;
743
```

# SWC-101 | ARITHMETIC OPERATION "\*" DISCOVERED

LINE 739

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
738 uint256 private constant _LPAllocation = 25 * 10**15 * 10**9;
739 uint256 private constant _burnAllocation = 35 * 10**15 * 10**9;
740
741 IUniswapV2Router02 public immutable uniswapV2Router;
742 address public immutable uniswapV2Pair;
743
```

# SWC-101 | ARITHMETIC OPERATION "\*\*" DISCOVERED

LINE 739

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
738 uint256 private constant _LPAllocation = 25 * 10**15 * 10**9;
739 uint256 private constant _burnAllocation = 35 * 10**15 * 10**9;
740
741 IUniswapV2Router02 public immutable uniswapV2Router;
742 address public immutable uniswapV2Pair;
743
```



# SWC-101 | ARITHMETIC OPERATION "\*\*" DISCOVERED

LINE 739

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
738 uint256 private constant _LPAllocation = 25 * 10**15 * 10**9;  
739 uint256 private constant _burnAllocation = 35 * 10**15 * 10**9;  
740  
741 IUniswapV2Router02 public immutable uniswapV2Router;  
742 address public immutable uniswapV2Pair;  
743
```

# SWC-101 | ARITHMETIC OPERATION "\*" DISCOVERED

LINE 747

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
746
747  uint256 public _maxWalletHolding = 65 * 10**13 * 10**9;
748  uint256 private numTokensSellToAddToLiquidity = 15 * 10**13 * 10**9;
749
750  event MinTokensBeforeSwapUpdated(uint256 minTokensBeforeSwap);
751
```

# SWC-101 | ARITHMETIC OPERATION "\*" DISCOVERED

LINE 747

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
746
747  uint256 public _maxWalletHolding = 65 * 10**13 * 10**9;
748  uint256 private numTokensSellToAddToLiquidity = 15 * 10**13 * 10**9;
749
750  event MinTokensBeforeSwapUpdated(uint256 minTokensBeforeSwap);
751
```

# SWC-101 | ARITHMETIC OPERATION "\*\*" DISCOVERED

LINE 747

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
746
747  uint256 public _maxWalletHolding = 65 * 10**13 * 10**9;
748  uint256 private numTokensSellToAddToLiquidity = 15 * 10**13 * 10**9;
749
750  event MinTokensBeforeSwapUpdated(uint256 minTokensBeforeSwap);
751
```

# SWC-101 | ARITHMETIC OPERATION "\*\*" DISCOVERED

LINE 747

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
746
747  uint256 public _maxWalletHolding = 65 * 10**13 * 10**9;
748  uint256 private numTokensSellToAddToLiquidity = 15 * 10**13 * 10**9;
749
750  event MinTokensBeforeSwapUpdated(uint256 minTokensBeforeSwap);
751
```

# SWC-101 | ARITHMETIC OPERATION "\*" DISCOVERED

LINE 748

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
747 uint256 public _maxWalletHolding = 65 * 10**13 * 10**9;  
748 uint256 private numTokensSellToAddToLiquidity = 15 * 10**13 * 10**9;  
749  
750 event MinTokensBeforeSwapUpdated(uint256 minTokensBeforeSwap);  
751 event SwapAndLiquifyEnabledUpdated(bool enabled);  
752
```

# SWC-101 | ARITHMETIC OPERATION "\*" DISCOVERED

LINE 748

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
747 uint256 public _maxWalletHolding = 65 * 10**13 * 10**9;  
748 uint256 private numTokensSellToAddToLiquidity = 15 * 10**13 * 10**9;  
749  
750 event MinTokensBeforeSwapUpdated(uint256 minTokensBeforeSwap);  
751 event SwapAndLiquifyEnabledUpdated(bool enabled);  
752
```

# SWC-101 | ARITHMETIC OPERATION "\*\*" DISCOVERED

LINE 748

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
747 uint256 public _maxWalletHolding = 65 * 10**13 * 10**9;  
748 uint256 private numTokensSellToAddToLiquidity = 15 * 10**13 * 10**9;  
749  
750 event MinTokensBeforeSwapUpdated(uint256 minTokensBeforeSwap);  
751 event SwapAndLiquifyEnabledUpdated(bool enabled);  
752
```



# SWC-101 | ARITHMETIC OPERATION "\*\*" DISCOVERED

LINE 748

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
747 uint256 public _maxWalletHolding = 65 * 10**13 * 10**9;  
748 uint256 private numTokensSellToAddToLiquidity = 15 * 10**13 * 10**9;  
749  
750 event MinTokensBeforeSwapUpdated(uint256 minTokensBeforeSwap);  
751 event SwapAndLiquifyEnabledUpdated(bool enabled);  
752
```

# SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 781

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
780  _rOwned[_burnPool] = _burnAllocation.mul(currentRate);
781  _rOwned[_msgSender()] = (_LPAllocation + _privateSaleAllocation).mul(currentRate);
782
783  emit Transfer(address(0), _msgSender(), _tTotal);
784  emit Transfer(_msgSender(), _dev, _devAllocation);
785
```

# SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 928

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
927
928   for (uint8 i = 0; i < recipients.length; i++) {
929     if (!recipients[i].isContract()) {
930       uint256 _rAlloc = allocations[i].mul(10**18).mul(currentRate);
931       _rOwned[recipients[i]] = _rAlloc;
932     }
```

# SWC-101 | ARITHMETIC OPERATION "\*\*" DISCOVERED

LINE 930

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
929  if (!recipients[i].isContract()) {
930  uint256 _rAlloc = allocations[i].mul(10**18).mul(currentRate);
931  _rOwned[recipients[i]] = _rAlloc;
932  _rOwned[_msgSender()] = _rOwned[_msgSender()] - _rAlloc;
933  emit Transfer(_msgSender(), recipients[i], allocations[i].mul(10**18));
934
```

# SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 932

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
931  _rOwned[recipients[i]] = _rAlloc;  
932  _rOwned[_msgSender()] = _rOwned[_msgSender()] - _rAlloc;  
933  emit Transfer(_msgSender(), recipients[i], allocations[i].mul(10**18));  
934  }  
935  }  
936
```

# SWC-101 | ARITHMETIC OPERATION "\*\*" DISCOVERED

LINE 933

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
932  _rOwned[_msgSender()] = _rOwned[_msgSender()] - _rAlloc;  
933  emit Transfer(_msgSender(), recipients[i], allocations[i].mul(10**18));  
934  }  
935  }  
936  }  
937
```

## SWC-101 | ARITHMETIC OPERATION "\*\*" DISCOVERED

LINE 1007

### low SEVERITY

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

### Source File

- eSHILLTOKEN.sol

### Locations

```
1006     return _amount.mul(_taxFee).div(  
1007         10**2  
1008     );  
1009 }  
1010  
1011
```

## SWC-101 | ARITHMETIC OPERATION "\*\*" DISCOVERED

LINE 1013

### low SEVERITY

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

### Source File

- eSHILLTOKEN.sol

### Locations

```
1012     return _amount.mul(_liquidityFee).div(  
1013         10**2  
1014     );  
1015 }  
1016  
1017
```



## SWC-101 | ARITHMETIC OPERATION "\*\*" DISCOVERED

LINE 1019

### low SEVERITY

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

### Source File

- eSHILLTOKEN.sol

### Locations

```
1018     return _amount.mul(_operationsFee).div(  
1019         10**2  
1020     );  
1021 }  
1022  
1023
```

# SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1095

## low SEVERITY

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

## Source File

- eSHILLTOKEN.sol

## Locations

```
1094   if(!_isExcludedFromFee[from] && !_isExcludedFromFee[to]){
1095   if (_lastBuyTime[from] != 0 && (_lastBuyTime[from] + (24 hours) > block.timestamp)
) {
1096   //increasing sell tax when user is selling within 24 hrs (Day Trader Tax)
1097   tax_multiplier = _dayTraderMultiplier;
1098   }
1099
```

## SWC-103 | A FLOATING PRAGMA IS SET.

LINE 6

### low SEVERITY

The current pragma Solidity directive is `""^0.6.12""`. 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

- eSHILLTOKEN.sol

### Locations

```
5 // SPDX-License-Identifier: Unlicensed
6 pragma solidity ^0.6.12;
7
8 interface IERC20 {
9
10
```

## SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 744

### 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

- eSHILLTOKEN.sol

### Locations

```
743
744 bool inSwapAndLiquify;
745 bool public swapAndLiquifyEnabled = true;
746
747 uint256 public _maxWalletHolding = 65 * 10**13 * 10**9;
748
```

## SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 929

### low SEVERITY

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

### Source File

- eSHILLTOKEN.sol

### Locations

```
928   for (uint8 i = 0; i < recipients.length; i++) {
929     if (!recipients[i].isContract()) {
930       uint256 _rAlloc = allocations[i].mul(10**18).mul(currentRate);
931       _rOwned[recipients[i]] = _rAlloc;
932       _rOwned[_msgSender()] = _rOwned[_msgSender()] - _rAlloc;
933     }
```

## SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 930

### low SEVERITY

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

### Source File

- eSHILLTOKEN.sol

### Locations

```
929  if (!recipients[i].isContract()) {
930  uint256 _rAlloc = allocations[i].mul(10**18).mul(currentRate);
931  _rOwned[recipients[i]] = _rAlloc;
932  _rOwned[_msgSender()] = _rOwned[_msgSender()] - _rAlloc;
933  emit Transfer(_msgSender(), recipients[i], allocations[i].mul(10**18));
934
```

## SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 931

### low SEVERITY

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

### Source File

- eSHILLTOKEN.sol

### Locations

```
930  uint256 _rAlloc = allocations[i].mul(10**18).mul(currentRate);
931  _rOwned[recipients[i]] = _rAlloc;
932  _rOwned[_msgSender()] = _rOwned[_msgSender()] - _rAlloc;
933  emit Transfer(_msgSender(), recipients[i], allocations[i].mul(10**18));
934  }
935
```

## SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 933

### low SEVERITY

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

### Source File

- eSHILLTOKEN.sol

### Locations

```
932  _rOwned[_msgSender()] = _rOwned[_msgSender()] - _rAlloc;  
933  emit Transfer(_msgSender(), recipients[i], allocations[i].mul(10**18));  
934  }  
935  }  
936  }  
937
```



## SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 933

### low SEVERITY

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

### Source File

- eSHILLTOKEN.sol

### Locations

```
932  _rOwned[_msgSender()] = _rOwned[_msgSender()] - _rAlloc;  
933  emit Transfer(_msgSender(), recipients[i], allocations[i].mul(10**18));  
934  }  
935  }  
936  }  
937
```

## SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 954

### low SEVERITY

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

### Source File

- eSHILLTOKEN.sol

### Locations

```
953  uint256[4] memory tValues = _getTValuesArray(tAmount);
954  uint256[3] memory rValues = _getRValuesArray(tAmount, tValues[1], tValues[2],
tValues[3]);
955  return (rValues[0], rValues[1], rValues[2], tValues[0], tValues[1], tValues[2],
tValues[3]);
956  }
957
958
```

## SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 954

### low SEVERITY

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

### Source File

- eSHILLTOKEN.sol

### Locations

```
953  uint256[4] memory tValues = _getTValuesArray(tAmount);
954  uint256[3] memory rValues = _getRValuesArray(tAmount, tValues[1], tValues[2],
tValues[3]);
955  return (rValues[0], rValues[1], rValues[2], tValues[0], tValues[1], tValues[2],
tValues[3]);
956  }
957
958
```

## SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 954

### low SEVERITY

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

### Source File

- eSHILLTOKEN.sol

### Locations

```
953  uint256[4] memory tValues = _getTValuesArray(tAmount);
954  uint256[3] memory rValues = _getRValuesArray(tAmount, tValues[1], tValues[2],
tValues[3]);
955  return (rValues[0], rValues[1], rValues[2], tValues[0], tValues[1], tValues[2],
tValues[3]);
956  }
957
958
```

## SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 955

### low SEVERITY

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

### Source File

- eSHILLTOKEN.sol

### Locations

```
954  uint256[3] memory rValues = _getRValuesArray(tAmount, tValues[1], tValues[2],
    tValues[3]);
955  return (rValues[0], rValues[1], rValues[2], tValues[0], tValues[1], tValues[2],
    tValues[3]);
956  }
957
958  function _getTValuesArray(uint256 tAmount) private view returns (uint256[4] memory
    val) {
959
```

## SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 955

### low SEVERITY

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

### Source File

- eSHILLTOKEN.sol

### Locations

```
954  uint256[3] memory rValues = _getRValuesArray(tAmount, tValues[1], tValues[2],
    tValues[3]);
955  return (rValues[0], rValues[1], rValues[2], tValues[0], tValues[1], tValues[2],
    tValues[3]);
956  }
957
958  function _getTValuesArray(uint256 tAmount) private view returns (uint256[4] memory
    val) {
959
```

## SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 955

### low SEVERITY

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

### Source File

- eSHILLTOKEN.sol

### Locations

```
954  uint256[3] memory rValues = _getRValuesArray(tAmount, tValues[1], tValues[2],
tValues[3]);
955  return (rValues[0], rValues[1], rValues[2], tValues[0], tValues[1], tValues[2],
tValues[3]);
956  }
957
958  function _getTValuesArray(uint256 tAmount) private view returns (uint256[4] memory
val) {
959
```

## SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 955

### low SEVERITY

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

### Source File

- eSHILLTOKEN.sol

### Locations

```
954  uint256[3] memory rValues = _getRValuesArray(tAmount, tValues[1], tValues[2],
tValues[3]);
955  return (rValues[0], rValues[1], rValues[2], tValues[0], tValues[1], tValues[2],
tValues[3]);
956  }
957
958  function _getTValuesArray(uint256 tAmount) private view returns (uint256[4] memory
val) {
959
```



## SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 955

### low SEVERITY

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

### Source File

- eSHILLTOKEN.sol

### Locations

```
954  uint256[3] memory rValues = _getRValuesArray(tAmount, tValues[1], tValues[2],
    tValues[3]);
955  return (rValues[0], rValues[1], rValues[2], tValues[0], tValues[1], tValues[2],
    tValues[3]);
956  }
957
958  function _getTValuesArray(uint256 tAmount) private view returns (uint256[4] memory
    val) {
959
```

## SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 955

### low SEVERITY

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

### Source File

- eSHILLTOKEN.sol

### Locations

```
954  uint256[3] memory rValues = _getRValuesArray(tAmount, tValues[1], tValues[2],
    tValues[3]);
955  return (rValues[0], rValues[1], rValues[2], tValues[0], tValues[1], tValues[2],
    tValues[3]);
956  }
957
958  function _getTValuesArray(uint256 tAmount) private view returns (uint256[4] memory
    val) {
959
```

## SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 955

### low SEVERITY

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

### Source File

- eSHILLTOKEN.sol

### Locations

```
954  uint256[3] memory rValues = _getRValuesArray(tAmount, tValues[1], tValues[2],
tValues[3]);
955  return (rValues[0], rValues[1], rValues[2], tValues[0], tValues[1], tValues[2],
tValues[3]);
956  }
957
958  function _getTValuesArray(uint256 tAmount) private view returns (uint256[4] memory
val) {
959
```

## SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1143

### low SEVERITY

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

### Source File

- eSHILLTOKEN.sol

### Locations

```
1142     address[] memory path = new address[](2);
1143     path[0] = address(this);
1144     path[1] = uniswapV2Router.WETH();
1145
1146     _approve(address(this), address(uniswapV2Router), tokenAmount);
1147
```

## SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1144

### low SEVERITY

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

### Source File

- eSHILLTOKEN.sol

### Locations

```
1143 path[0] = address(this);
1144 path[1] = uniswapV2Router.WETH();
1145
1146 _approve(address(this), address(uniswapV2Router), tokenAmount);
1147
1148
```

# 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.