HuhToken

Smart Contract Security Audit

Prepared by ShellBoxes

November 6th, 2021 - November 15th, 2021

Shellboxes.com

contact@shellboxes.com

Document Properties

Client	HuhToken
Version	1.0
Classification	Public

Scope

The HuhToken Contract in the HuhToken Repository

Files	MD5 Hash
HuhToken-30-10-21.sol	c48fcab7f562a2a73a80b23face52dd0
RewardDistributor.sol	68297bc211f38746f84347c7b30ae233

Contacts

COMPANY	EMAIL
ShellBoxes	contact@shellboxes.com

Contents

1 Introduction

HuhToken engaged ShellBoxes to conduct a security assessment on the HuhToken beginning on November 6th, 2021 and ending November 15th, 2021. In this report, we detail our methodical approach to evaluate potential security issues associated with the implementation of smart contracts, by exposing possible semantic discrepancies between the smart contract code and design document, and by recommending additional ideas to optimize the existing code. Our findings indicate that the current version of smart contracts can still be enhanced further due to the presence of many security and performance concerns.

This document summarizes the findings of our audit.

1.1 About HuhToken

UH has an intelligent referral system that anyone can use, whether you are a beginner to the crypto scene or a well-seasoned pro, this referral system threatens to change the whole game. This one-of-a-kind referral system allows you to earn cold hard cash straight into your wallet through BNB. Refer anyone and everyone to earn 10% BNB from their first purchase. The next best part is you and your referral both receive a discount in selling tax from 20% to 10%. They have created a new cryptocurrency that uses blockchain technology to give everyone a chance to gain unique rewards. Their mission is to challenge the status-quo and create a source of income for everyone. The smart contract will re-distribute both a portion of BNB and HUH on every purchase and sale to the holders

Issuer	HuhToken
Website	`https://huh.social/`
Type	Solidity Smart Contract
Audit Method	Whitebox

1.2 Approach & Methodology

ShellBoxes used a combination of manual and automated security testing to achieve a balance between efficiency, timeliness, practicability, and correctness within the audit's

scope. While manual testing is advised for identifying problems in logic, procedure, and implementation, automated testing techniques help to expand the coverage of smart contracts and can quickly detect code that does not comply with security best practices.

1.2.1 Risk Methodology

Vulnerabilities or bugs identified by ShellBoxes are ranked using a risk assessment technique that considers both the LIKELIHOOD and IMPACT of a security incident. This framework is effective at conveying the features and consequences of technological vulnerabilities.

Its quantitative paradigm enables repeatable and precise measurement, while also revealing the underlying susceptibility characteristics that were used to calculate the Risk scores. A risk level will be assigned to each vulnerability on a scale of 5 to 1, with 5 indicating the greatest possibility or impact.

- Likelihood quantifies the probability of a certain vulnerability being discovered and exploited in the untamed.
- Impact quantifies the technical and economic costs of a successful attack.
- Severity indicates the risk's overall criticality.

Probability and impact are classified into three categories: H, M, and L, which correspond to high, medium, and low, respectively. Severity is determined by probability and impact and is categorized into four levels, namely Critical, High, Medium, and Low.

Impact				
	High	Critical	High	Medium
	Medium	High	Medium	Low
	Low	Medium	Low	Low
		High	Medium	Low

Likelihood

2 Findings Overview

2.1 Summary

The following is a synopsis of our conclusions from our analysis of the HuhToken implementation. During the first part of our audit, we examine the smart contract source code and run the codebase via a static code analyzer. The objective here is to find known coding problems statically and then manually check (reject or confirm) issues highlighted by the tool. Additionally, we check business logics, system processes, and DeFi-related components manually to identify potential hazards and/or defects.

2.2 Key Findings

In general, these smart contracts are well-designed and constructed, but their implementation might be improved by addressing the discovered flaws, which include , 3 high-severity, 6 medium-severity, 6 low-severity vulnerabilities.

Vulnerabilities	Severity	Status
Front Run in Register Code	HIGH	Acknowledged
Usage Of transfer Instead Of safeTransfer	HIGH	Fixed
Usage Of transfer Instead Of safeTransfer	HIGH	Fixed
Catching A Revert	MEDIUM	Fixed
Race Condition in Variables	MEDIUM	Acknowledged
Approve Race	MEDIUM	Acknowledged
For Loop Over Dynamic Array	MEDIUM	Acknowledged
Old marketingWallet/rewardDistrubtor are not included In Fee	MEDIUM	Fixed
Catching a revert	MEDIUM	Fixed
Usage of Block.TimeStamp	LOW	Acknowledged
Owner Can Renounce Ownership	LOW	Acknowledged
Missing Value Verification	LOW	Fixed
Division Before Multiplication	LOW	Fixed
Usage of block.TimeStamp	LOW	Acknowledged

Missing Address Verification	LOW	Acknowledged

3 Finding Details

A HuhToken.sol

A.1 Front Run in Register Code [HIGH]

Description:

The registerCode function is vulnerable to a front run vulnerability, when a user calls registerCode an attacker might watch the pending transactions to see the code provided and then calls the registerCode function using the same code but with a higher gas which make him a better choice for validators, thus the attacker's transaction will be executed first while the user's transaction will get reverted.

Code:

Listing 1: HuhToken.sol

```
1174 function registerCode(string memory code) external {
1175     bytes memory code_ = bytes(code);
1176     require(code_.length > 0, "Invalid code!");
1177     require(referUserForCode[code_] == address(0), "Code already used!")
         ↪ ;
1178     require(
1179         referCodeForUser[msg.sender].length == 0,
1180         "User already generated code!"
1181     );
1182     _registerCode(msg.sender, code_);
1183 }
```

Risk Level:

Likelihood – 3
Impact - 5

Recommendation:

The Team can combine the code provider by the user with the address of caller using hashing functions or concatenation to prevent this kind of attacks.

Status - Acknowledged

The Team has acknowledged the risk.

A.2 Usage Of transfer Instead Of safeTransfer [HIGH]

Description:

The ERC20 standard token implementation functions return the transaction status as a boolean.It is a good practice to check for the return status of the function call to ensure that the transaction was successful. It is the developer's responsibility to enclose these function calls with require() to ensure that, when the intended ERC20 function call returns false, the caller transaction also fails. However, it is mostly missed by developers when they carry out checks in effect, the transaction would always succeed, even if the token transfer did not.

Code:

Listing 2: HuhToken.sol

```
2101 function withdrawTokens(address token, address recipient)
2102     external
2103     onlyOwner
2104 {
2105     require(token != address(0), "Invalid Token!");
2106     require(recipient != address(0), "Invalid Recipient!");
2107     uint256 balance = IBEP20(token).balanceOf(address(this));
2108     if (balance > 0) {
2109         IBEP20(token).transfer(recipient, balance);
2110     }
2111 }
```

Risk Level:

Likelihood – 2
Impact - 5

Recommendation:

Use the safeTransfer function from the safeERC20 Implementation, or put the transfer call inside an assert or require to verify that it returned true.

Listing 3: HuhToken.sol

```
2101 function withdrawTokens(address token, address recipient) external
       ↪ onlyOwner{
2102     require(token != address(0), "Invalid Token!");
2103     require(recipient != address(0), "Invalid Recipient!");
2104     uint256 balance = IBEP20(token).balanceOf(address(this));
2105     if (balance > 0) {
2106         require(IBEP20(token).transfer(recipient, balance)), "Transfer
               ↪ Failed");
2107     }
2108 }
```

Status - Fixed

The Team has fixed the issue by adding a require statement which verifies that the transaction has passed successfully.

A.3 Catching A Revert [MEDIUM]

Description:

In the _sendToRewardDistributor function, there is a call to the addRewardHolderShare function from the rewardDistributor contract. This call is inside a try catch block, if the transaction fails the contract will ignore the error, while this can hurt the business logic in terms of reward distribution.

Code:

Listing 4: HuhToken.sol

```
167 function _sendToRewardDistributor(address sender, address
        ↪ rewardRecipient, uint256 tAmount, uint256 rAmount) private {
168     _rOwned[address(rewardDistributor)] = _rOwned[address(
            ↪ rewardDistributor)].add(rAmount);
169     if (_isExcluded[address(rewardDistributor)])
170         _tOwned[address(rewardDistributor)] = _tOwned[address(
                ↪ rewardDistributor)].add(tAmount);
171     emit Transfer(sender, address(rewardDistributor), tAmount);
172     try
173         rewardDistributor.addRewardHolderShare(rewardRecipient, tAmount)
                ↪ /
174     {} catch {}
175     userReferralReward[rewardRecipient] = userReferralReward[
            ↪ rewardRecipient].add(tAmount);
176     totalReferralReward = totalReferralReward.add(tAmount);
177 }
```

Risk Level:

Likelihood – 2

Impact - 3

Recommendation:

The team should remove the try catch block or in the catch revert.

Listing 5: HuhToken (Line 1674)

```
167 function _sendToRewardDistributor(address sender, address
        ↪ rewardRecipient, uint256 tAmount, uint256 rAmount) private {
168     _rOwned[address(rewardDistributor)] = _rOwned[address(
            ↪ rewardDistributor)].add(rAmount);
169     if (_isExcluded[address(rewardDistributor)])
```

```
170         _tOwned[address(rewardDistributor)] = _tOwned[address(
                ↪ rewardDistributor)].add(tAmount);
171     emit Transfer(sender, address(rewardDistributor), tAmount);
172     rewardDistributor.addRewardHolderShare(rewardRecipient, tAmount)
173     userReferralReward[rewardRecipient] = userReferralReward[
            ↪ rewardRecipient].add(tAmount);
174     totalReferralReward = totalReferralReward.add(tAmount);
175 }
```

Status - Fixed

The Team has fixed the issue by removing the try catch block.

A.4 Race Condition in Variables [MEDIUM]

Description:

Multiple variables in the smart contact have a setter. If the user checks the value of this variable, then calls one of the functions that uses it, and the owner updates the one of these variables, the order of the transaction might overturn and the user's transaction in this case will be executed with the new variable value without him knowing about it.

Code:

Listing 6: HuhToken.sol

```
1480 function _normalBuy(address sender, address recipient, uint256 amount)
       ↪ private {
1481     uint256 currentRate = _getRate();
1482     uint256 rAmount = amount.mul(currentRate);
1483     uint256 rLiquidityFee = amount.div(100).mul(liquidityFeeOnBuy).mul(
            ↪ currentRate);
1484     uint256 rHuhdistributionFee = amount.div(100).mul(
            ↪ HuHdistributionFeeOnBuy).mul(currentRate);
```

```
1485     uint256 rMarketingFee = amount.div(100).mul(marketingFeeOnBuy).mul(
             ↪ currentRate);
```

Listing 7: HuhToken.sol

```
1525 function _whitelistedBuy(address sender, address recipient, uint256
         ↪ amount) private {
1526     uint256 currentRate = _getRate();
1527     uint256 rAmount = amount.mul(currentRate);
1528     uint256 tReferralRewardAmount = amount.div(100).mul(referralReward);
1529     uint256 rReferralRewardAmount = tReferralRewardAmount.mul(
             ↪ currentRate);
1530     uint256 rLiquidityFee = amount.div(100).mul(
             ↪ liquidityFeeOnWhiteListedBuy).mul(currentRate);
1531     uint256 rHuhdistributionFee = amount.div(100).mul(
             ↪ HuHdistributionFeeOnBuyWhiteListed).mul(currentRate);
1532     uint256 rMarketingFee = amount.div(100).mul(
             ↪ marketingFeeOnBuyWhiteListed).mul(currentRate);
```

Listing 8: HuhToken.sol

```
1582 function _normalSell(address sender, address recipient, uint256 amount)
         ↪ private {
1583     uint256 currentRate = _getRate();
1584     uint256 rAmount = amount.mul(currentRate);
1585     uint256 rLiquidityFee = amount.div(100).mul(liquidityFeeOnSell).mul(
             ↪ currentRate);
1586     uint256 rHuhdistributionFee = amount.div(100).mul(
             ↪ HuHdistributionFeeOnSell).mul(currentRate);
1587     uint256 rMarketingFee = amount.div(100).mul(marketingFeeOnSell).mul(
             ↪ currentRate)
```

Listing 9: HuhToken.sol

```
1627 function _whitelistedSell(address sender, address recipient, uint256
         ↪ amount) private {
1628     uint256 currentRate = _getRate();
```

```
1629      uint256 rAmount = amount.mul(currentRate);
1630      uint256 rLiquidityFee = amount.div(100).mul(
              ↪ liquidityFeeOnWhiteListedSell).mul(currentRate);
1631      uint256 rHuhdistributionFee = amount.div(100).mul(
              ↪ HuHdistributionFeeOnWhiteListedSell).mul(currentRate);
1632      uint256 rMarketingFee = amount.div(100).mul(
              ↪ marketingFeeOnWhiteListedSell).mul(currentRate);
```

Risk Level:

Likelihood – 2

Impact - 3

Recommendation:

Add the concerned variable in the arguments of the called function, then add require statements that verifies that the value provided in the arguments is the same as the one that is stored in the smart contract.

Listing 10: HuhToken.sol

```
1480  function _normalBuy(address sender, address recipient, uint256 amount,
          ↪ uint8 _liquidityFeeOnBuy,uint8 _HuHdistributionFeeOnBuy, uint8
          ↪ _marketingFeeOnBuy) private {
1481      require(_liquidityFeeOnBuy!= liquidityFeeOnBuy, "Invalid fee value
              ↪ !");
1482      require(_HuHdistributionFeeOnBuy != HuHdistributionFeeOnBuy), "
              ↪ Invalid fee value!");
1483      require(_marketingFeeOnBuy != marketingFeeOnBuy " Invalid fee value
              ↪ !");
1484      uint256 currentRate = _getRate();
1485      uint256 rAmount = amount.mul(currentRate);
1486      uint256 rLiquidityFee = amount.div(100).mul(liquidityFeeOnBuy).mul(
              ↪ currentRate);
1487      uint256 rHuhdistributionFee = amount.div(100).mul(
              ↪ HuHdistributionFeeOnBuy).mul(currentRate);
```

```
1488      uint256 rMarketingFee = amount.div(100).mul(marketingFeeOnBuy).mul(
              ↪ currentRate);
```

Status - Acknowledged

The Team has acknowledged the risk.

A.5 Approve Race [MEDIUM]

Description:

The standard ERC20 implementation contains a widely-known racing condition in it approve function, wherein a spender is able to witness the token owner broadcast a transaction altering their approval and quickly sign and broadcast a transaction using transferFrom to move the current approved amount from the owner's balance to the spender. If the spender's transaction is validated before the owner's, the spender will be able to get both approval amounts of both transactions.

Code:

Listing 11: HuhToken.sol

```
1402  function _approve(address owner, address spender, uint256 amount)
          ↪ private {
1403      require(owner != address(0), "BEP20: approve from the zero address")
              ↪ ;
1404      require(spender != address(0), "BEP20: approve to the zero address")
              ↪ ;
1405      _allowances[owner][spender] = amount;
1406      emit Approval(owner, spender, amount);
1407  }
```

Risk Level:

Likelihood – 2

Impact - 3

Recommendation:

Use increaseAllowance and decreaseAllowance functions to modify the approval amount instead of using the approve function to modify it.

Status - Acknowledged

The team has acknowledged the risk.

A.6 For Loop Over Dynamic Array [MEDIUM]

Description:

When smart contracts are deployed or their associated functions are invoked, the execution of these operations always consumes a certain quantity of gas, according to the amount of computation required to accomplish them. Modifying an unknown-size array that grows in size over time can result in a Denial-of-Service. Simply by having an excessively huge array, users can exceed the gas limit, therefore preventing the transaction from ever succeeding.

Code:

Listing 12: HuhToken.sol

```
1382 function _getCurrentSupply() private view returns (uint256, uint256) {
1383     uint256 rSupply = _rTotal;
1384     uint256 tSupply = _tTotal;
1385     for (uint256 i = 0; i < _excluded.length; i++) {
1386         if (_rOwned[_excluded[i]] > rSupply _tOwned[_excluded[i]] >
             ↪ tSupply)
1387             return (_rTotal, _tTotal);
1388         rSupply = rSupply.sub(_rOwned[_excluded[i]]);
1389         tSupply = tSupply.sub(_tOwned[_excluded[i]]);
1390     }
1391     if (rSupply < _rTotal.div(_tTotal)) {
1392         return (_rTotal, _tTotal);
```

```
1393        }
1394        return (rSupply, tSupply);
1395    }
```

Listing 13: HuhToken.sol

```
1769  function _includeInReflection(address account) private {
1770      require(_isExcluded[account], "Account is already included");
1771      for (uint256 i = 0; i < _excluded.length; i++) {
1772          if (_excluded[i] == account) {
1773              _excluded[i] = _excluded[_excluded.length - 1];
1774              _rOwned[account] = reflectionFromToken(_tOwned[account]);
1775              _isExcluded[account] = false;
1776              _excluded.pop();
1777              break;
1778          }
1779      }
1780  }
```

Risk Level:

Likelihood – 1
Impact - 3

Recommendation:

Avoid actions that involve looping across the entire data structure. If you really must loop over an array of unknown size, arrange for it to consume many blocs and thus multiple transactions.

Status - Acknowledged

The team has acknowledged the risk.

A.7 Old marketingWallet/rewardDistrubtor are not included In Fee [MEDIUM]

Description:

When setting the marketingWallet and rewardDistrubtor addresses, the old addresses are not included back in the fee. Thus, all the old addresses will be excluded from the fee transactions.

Code:

Listing 14: HuhToken.sol

```
1066 function updateMarketingWallet(address _marketingWallet) external
         ↪ onlyOwner {
1067     require(_marketingWallet != address(0), "Zero address not allowed!")
             ↪ ;
1068     marketingWallet = _marketingWallet;
1069     emit UpdateMarketingWallet(_marketingWallet);
1070 }
```

Listing 15: HuhToken.sol

```
1115 function updateRewardDistributor(address _rewardDistributor) external
         ↪ onlyOwner {
1116     require(address(rewardDistributor) != _rewardDistributor, "Reward
             ↪ Distributor already exists!");
1117     rewardDistributor = IRewardDistributor(_rewardDistributor);
1118     _allowances[address(this)][address(rewardDistributor)] = _MAX;
1119     _allowances[address(rewardDistributor)][address(pcsV2Router)] = _MAX
             ↪ ;
1120     _isExcludedFromFee[address(rewardDistributor)] = true;
1121     _excludeFromReflection(address(rewardDistributor));
1122     emit UpdateRewardDistributor(_rewardDistributor);
1123 }
```

Risk Level:

Likelihood – 3

Impact - 3

Recommendation:

Include the old value of the address to the fee before updating it to the new value using the mapping `_isExcludedFromFee` and set it back to `false`.

Listing 16: HuhToken (Line 1066)

```
1066 function updateMarketingWallet(address _marketingWallet) external
         ↪ onlyOwner {
1067     require(_marketingWallet != address(0), "Zero address not allowed!")
             ↪ ;
1068     _isExcludedFromFee[marketingWallet] = false;
1069     marketingWallet = _marketingWallet;
1070     emit UpdateMarketingWallet(_marketingWallet);
1071 }
```

Listing 17: HuhToken.sol

```
1115 function updateRewardDistributor(address _rewardDistributor) external
         ↪ onlyOwner {
1116     require(address(rewardDistributor) != _rewardDistributor, "Reward
             ↪ Distributor already exists!");
1117     _isExcludedFromFee[rewardDistributor] = false;
1118     rewardDistributor = IRewardDistributor(_rewardDistributor);
1119     _allowances[address(this)][address(rewardDistributor)] = _MAX;
1120     _allowances[address(rewardDistributor)][address(pcsV2Router)] = _MAX
             ↪ ;
1121     _isExcludedFromFee[address(rewardDistributor)] = true;
1122     _excludeFromReflection(address(rewardDistributor));
1123     emit UpdateRewardDistributor(_rewardDistributor);
```

Status - Fixed

The team has fixed the issue by setting back `_isExcludedFromFee` for the old address to `false`.

A.8 Usage of Block.TimeStamp [LOW]

Description:

`block.timestamp` is used in the contract. The variable block is a set of variables. The timestamp does not always reflect the current time and may be inaccurate. The value of a block can be influenced by miners. Maximal Extractable Value attacks require a timestamp of up to 900 seconds. There is no guarantee that the value is right, all what is guaranteed is that it is higher than the timestamp of the previous block.

Code:

Listing 18: HuhToken.sol

```
1720 function _swapAndAddToLiquidity() private swapping {
1721     uint256 tokenAmountForLiquidity =
         ↪ amountOfTokensToAddToLiquidityThreshold;
1722     uint256 amountToSwap = tokenAmountForLiquidity.div(2);
1723     uint256 amountAnotherHalf = tokenAmountForLiquidity.sub(amountToSwap
         ↪ );
1724     address[] memory path = new address[](2);
1725     path[0] = address(this);
1726     path[1] = pcsV2Router.WETH();
1727     uint256 balanceBefore = address(this).balance;
1728     pcsV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(
         ↪ amountToSwap, 0,
1729     path, address(this), block.timestamp.add(30));
1730     uint256 differenceBnb = address(this).balance.sub(balanceBefore);
1731     pcsV2Router.addLiquidityETH{value: differenceBnb}(address(this),
         ↪ amountAnotherHalf,0, 0, _DEAD_ADDRESS, block.timestamp.add(30)
         ↪ );
```

```
1732        emit SwapAndLiquify(differenceBnb, amountToSwap);
1733    }
```

Risk Level:

Likelihood – 2
Impact - 2

Recommendation:

You can use an Oracle to get the exact time or verify if a delay of 900 seconds won't destroy the logic of the contract.

Status - Acknowledged

The team has acknowledged the risk

A.9 Owner Can Renounce Ownership [LOW]

Description:

Typically, the contract's owner is the account that deploys the contract. As a result, the owner is able to perform certain privileged activities on his behalf. The renounceOwnership function is used in smart contracts to renounce ownership. Otherwise, if the contract's ownership has not been transferred previously, it will never have an Owner, which is risky.

Code:

Listing 19: HuhToken.sol

```
790 contract HuhToken is Context, IBEP20, Ownable {
791     using SafeMath for uint256;
```

Risk Level:

Likelihood – 1
Impact - 3

Recommendation:

It is advised that the Owner cannot call renounceOwnership without first transferring ownership to a different address. Additionally, if a multi-signature wallet is utilized, executing the renounceOwnership method for two or more users should be confirmed. Alternatively, the renounce ownership functionality can be disabled by overriding it.

Status - Acknowledged

The team has acknowledged the risk.

A.10 Missing Value Verification [LOW]

Description:

Certain functions lack a safety check in the values, the values of the arguments should include some safety checks test, otherwise, the contract's functionality may get hurt.

Code:

Listing 20: HuhToken.sol

```
996 function changeFeesForNormalBuy(uint8 _liquidityFeeOnBuy, uint8
        ↪ _marketingFeeOnBuy,
997     uint8 _HuHdistributionFeeOnBuy) external onlyOwner {
998     liquidityFeeOnBuy = _liquidityFeeOnBuy;
999     marketingFeeOnBuy = _marketingFeeOnBuy;
1000    HuHdistributionFeeOnBuy = _HuHdistributionFeeOnBuy;
1001    emit ChangeFeesForNormalBuy(_liquidityFeeOnBuy, _marketingFeeOnBuy,
            ↪ _HuHdistributionFeeOnBuy);
1002 }
```

Listing 21: HuhToken.sol

```
1012 function changeFeesForWhiteListedBuy(uint8 _liquidityFeeOnBuy, uint8
        ↪ _marketingFeeOnBuy,
1013     uint8 _HuHdistributionFeeOnBuy ) external onlyOwner {
1014     liquidityFeeOnWhiteListedBuy = _liquidityFeeOnBuy;
1015     marketingFeeOnBuyWhiteListed = _marketingFeeOnBuy;
1016     HuHdistributionFeeOnBuyWhiteListed = _HuHdistributionFeeOnBuy;
1017     emit ChangeFeesForWhiteListedBuy(_liquidityFeeOnBuy,
            ↪ _marketingFeeOnBuy, _HuHdistributionFeeOnBuy);
1018 }
```

Listing 22: HuhToken.sol

```
1028 function changeFeesForNormalSell(uint8 _liquidityFeeOnSell, uint8
        ↪ _marketingFeeOnSell,
1029     uint8 _HuHdistributionFeeOnSell) external onlyOwner {
1030     liquidityFeeOnSell = _liquidityFeeOnSell;
1031     marketingFeeOnSell = _marketingFeeOnSell;
1032     HuHdistributionFeeOnSell = _HuHdistributionFeeOnSell;
1033     emit ChangeFeesForNormalSell(_liquidityFeeOnSell,
            ↪ _marketingFeeOnSell, _HuHdistributionFeeOnSell);
1034 }
```

Listing 23: HuhToken.sol

```
1044 function changeFeesForWhitelistedSell(uint8 _liquidityFeeOnSell, uint8
        ↪ _marketingFeeOnSell,
1045     uint8 _HuHdistributionFeeOnSell) external onlyOwner {
1046     liquidityFeeOnWhiteListedSell = _liquidityFeeOnSell;
1047     marketingFeeOnWhiteListedSell = _marketingFeeOnSell;
1048     HuHdistributionFeeOnWhiteListedSell = _HuHdistributionFeeOnSell;
1049     emit ChangeFeesForWhitelistedSell(_liquidityFeeOnSell,
            ↪ _marketingFeeOnSell, _HuHdistributionFeeOnSell);
1050 }
```

Listing 24: HuhToken.sol

```
1060 function changeReferralReward(uint8 _referralReward) external onlyOwner
       ↪ {
1061     referralReward = _referralReward;
1062     emit ChangeReferralReward(_referralReward);
1063 }
```

Listing 25: HuhToken.sol

```
1149 function setMaxTxPercent(uint256 maxTxPercent) external onlyOwner {
1150     maxTxAmount = _tTotal.mul(maxTxPercent).div(10**2);
1151     emit SetMaxTxPercent(maxTxPercent);
1152 }
```

Listing 26: HuhToken.sol

```
1753 function _reflectFee(uint256 rFee, uint256 tFee) private {
1754     _rTotal = _rTotal.sub(rFee);
1755     _tFeeTotal = _tFeeTotal.add(tFee);
1756 }
```

Risk Level:

Likelihood – 1

Impact - 3

Recommendation:

It's recommended to undertake further validation prior to user-supplied data. The concerns can be resolved by utilizing require statements. For the fees, the value should be less than 100.

Listing 27: HuhToken.sol

```
996 function changeFeesForNormalBuy(uint8 _liquidityFeeOnBuy, uint8
      ↪ _marketingFeeOnBuy,
997     uint8 _HuHdistributionFeeOnBuy) external onlyOwner {
998     require(_liquidityFeeOnBuy < 100, "Fee should be less than 100!");
```

```
999     require(_HuHdistributionFeeOnBuy < 100, " Fee should be less than
            ↪ 100!");
1000    require(_marketingFeeOnBuy < 100 ,"Fee should be less than 100!");
1001    liquidityFeeOnBuy = _liquidityFeeOnBuy;
1002    marketingFeeOnBuy = _marketingFeeOnBuy;
1003    HuHdistributionFeeOnBuy = _HuHdistributionFeeOnBuy;
```

Status - Fixed

The team has fixed the issue as recommended by verifying the values coming from the arguments.

A.11 Division Before Multiplication [LOW]

Description:

Integer division in solidity may truncate. As a result, dividing before multiplying may result in a loss of precision. Due to precision's sensitivity, this may result in certain abnormalities in the contract's logic.

Code:

Listing 28: HuhToken.sol

```
1480 function _normalBuy(address sender, address recipient, uint256 amount)
        ↪ private {
1481    uint256 currentRate = _getRate();
1482    uint256 rAmount = amount.mul(currentRate);
1483    uint256 rLiquidityFee = amount.div(100).mul(liquidityFeeOnBuy).mul(
            ↪ currentRate);
1484    uint256 rHuhdistributionFee = amount.div(100).mul(
            ↪ HuHdistributionFeeOnBuy).mul(currentRate);
1485    uint256 rMarketingFee = amount.div(100).mul(marketingFeeOnBuy).mul(
            ↪ currentRate);
```

Listing 29: HuhToken.sol

```
1525 function _whitelistedBuy(address sender, address recipient, uint256
       ↪ amount) private {
1526     uint256 currentRate = _getRate();
1527     uint256 rAmount = amount.mul(currentRate);
1528     uint256 tReferralRewardAmount = amount.div(100).mul(referralReward);
1529     uint256 rReferralRewardAmount = tReferralRewardAmount.mul(
           ↪ currentRate);
1530     uint256 rLiquidityFee = amount.div(100).mul(
           ↪ liquidityFeeOnWhiteListedBuy).mul(currentRate);
1531     uint256 rHuhdistributionFee = amount.div(100).mul(
           ↪ HuHdistributionFeeOnBuyWhiteListed).mul(currentRate);
1532     uint256 rMarketingFee = amount.div(100).mul(
           ↪ marketingFeeOnBuyWhiteListed).mul(currentRate);
```

Listing 30: HuhToken.sol

```
1582 function _normalSell(address sender, address recipient, uint256 amount)
       ↪ private {
1583     uint256 currentRate = _getRate();
1584     uint256 rAmount = amount.mul(currentRate);
1585     uint256 rLiquidityFee = amount.div(100).mul(liquidityFeeOnSell).mul(
           ↪ currentRate);
1586     uint256 rHuhdistributionFee = amount.div(100).mul(
           ↪ HuHdistributionFeeOnSell).mul(currentRate);
1587     uint256 rMarketingFee = amount.div(100).mul(marketingFeeOnSell).mul(
           ↪ currentRate);
```

Risk Level:

Likelihood – 1

Impact - 2

Recommendation:

The Team should do the multiplication operations before the division operations.

Status - Fixed

The team has fixed the issue by doing the multiplication operations before the divisions.

B RewardDistributor.sol

B.1 Usage Of transfer Instead Of safeTransfer [HIGH]

Description:

The ERC20 standard token implementation functions also return the transaction status as a Boolean. It is a good practice to check for the return status of the function call to ensure that the transaction was successful. It is the developer's responsibility to enclose these function calls with require() to ensure that, when the intended ERC20 function call returns false, the caller transaction also fails. However, it is mostly missed by developers when they carry out checks; in effect, the transaction would always succeed, even if the token transfer did not.

Code:

Listing 31: RewardDistributor.sol

```
715 function setRewardTokenAddress(address _rewardToken) external onlyOwner
        ↪ {
716     require(rewardToken != _rewardToken && _rewardToken != address(0), "
            ↪ Invalid Reward Token!");
717     if (rewardToken != address(0)) {
718         uint256 balance = IERC20(rewardToken).balanceOf(address(this));
719             if (balance > 0) {
720                 IERC20(rewardToken).transfer(tokenOwner, balance);
721             }
722     }
```

```
723     rewardToken = _rewardToken;
724     emit SetRewardTokenAddress(_rewardToken);
725 }
```

Listing 32: RewardDistributor.sol

```
739 function upgradeDistributor(address newDistributor) external onlyOwner {
740     require(newDistributor != address(this) && newDistributor != address
        ↪ (0),"Invalid Distributor!");
741     uint256 balance = IERC20(rewardToken).balanceOf(address(this));
742     if (balance > 0) IERC20(rewardToken).transfer(newDistributor,
        ↪ balance);
743     emit UpgradeDistributor(newDistributor);
744     selfdestruct(payable(tokenOwner));
745 }
```

Listing 33: RewardDistributor.sol

```
802 function emergencyWithdrawTokens(address token) external onlyOwner {
803     require(token != address(0), "Invalid Token!");
804     uint256 balance = IERC20(token).balanceOf(address(this));
805     if (balance > 0) {
806         IERC20(token).transfer(tokenOwner, balance);
807     }
808 }
```

Risk Level:

Likelihood – 2
Impact - 5

Recommendation:

Use the safeTransfer function from the safeERC20 Implementation, or put the transfer call inside an assert or require to verify that it returned true.

Listing 34: RewardDistributor.sol

```
715 function setRewardTokenAddress(address _rewardToken) external onlyOwner
        ↪ {
716     require(rewardToken != _rewardToken && _rewardToken != address(0), "
            ↪ Invalid Reward Token!");
717     if (rewardToken != address(0)) {
718         uint256 balance = IERC20(rewardToken).balanceOf(address(this));
719             if (balance > 0) {
720                 require (IERC20(rewardToken).transfer(tokenOwner, balance)
                        ↪ , "Transfer Failed");
721             }
722     }
723     rewardToken = _rewardToken;
724     emit SetRewardTokenAddress(_rewardToken);
725 }
```

Status - Fixed

The Team has fixed the issue by adding a require statement which verifies that the transaction has passes successfully.

B.2 Catching a revert [MEDIUM]

Description:

In the _giftReward function, there is a call to the transfer function from the rewardToken ERC20 contract. This call will never revert because the transfer function only returns true or false. Also, the call is inside a try catch block, if the transaction fails the contract will ignore the error while this can hurt the business logic in terms of reward distribution.

Code:

Listing 35: RewardDistributor.sol

```
856 function _giftReward(address rewardRecipient, address giftRecipient,
        ↪ uint256 amount) private {
857     require(rewardRecipient != address(0), "Invalid Reward Recipient!");
858     require(giftRecipient != address(0), "Invalid Gift Recipient!");
859     require(referralShares[rewardRecipient].amount > 0, "Insufficient
            ↪ Balance!");
860     require(amount > 0, "Invalid Amount!");
861     require(amount <= referralShares[rewardRecipient].amount, "
            ↪ Insufficient Balance!");
862     if (referralShares[rewardRecipient].amount <=IERC20(rewardToken).
            ↪ balanceOf(address(this))) {
863         IERC20(rewardToken).approve(rewardRecipient, amount);
864         try IERC20(rewardToken).transfer(giftRecipient, amount) {} catch
                ↪ {}
865     }
866     referralShares[rewardRecipient].amount = referralShares[
            ↪ rewardRecipient]
867         .amount
868         .sub(amount);
869 }
```

Risk Level:

Likelihood – 2

Impact - 3

Recommendation:

The team should remove the try catch block or add a revert in the catch statement.

Listing 36: RewardDistributor.sol

```
856 function _giftReward(address rewardRecipient, address giftRecipient,
        ↪ uint256 amount) private {
857     require(rewardRecipient != address(0), "Invalid Reward Recipient!");
858     require(giftRecipient != address(0), "Invalid Gift Recipient!");
859     require(referralShares[rewardRecipient].amount > 0, "Insufficient
            ↪ Balance!");
860     require(amount > 0, "Invalid Amount!");
861     require(amount <= referralShares[rewardRecipient].amount, "
            ↪ Insufficient Balance!");
862     if (referralShares[rewardRecipient].amount <=IERC20(rewardToken).
            ↪ balanceOf(address(this))) {
863         IERC20(rewardToken).approve(rewardRecipient, amount);
864         require (IERC20(rewardToken).transfer(giftRecipient, amount),"
                ↪ Transfer Failed");
865     }
866     referralShares[rewardRecipient].amount = referralShares[
            ↪ rewardRecipient]
867         .amount
868         .sub(amount);
869 }
```

Status - Fixed

The Team has fixed the issue by removing the try catch block.

B.3 Usage of block.TimeStamp [LOW]

Description:

block.timestamp is used in the contract. The variable block is a set of variables. The timestamp does not always reflect the current time and may be inaccurate. The value of a block can be influenced by miners. Maximal Extractable Value attacks require a timestamp of up to 900 seconds. There is no guarantee that the value is right, all what is guaranteed is that it

is higher than the timestamp of the previous block.

Code:

Listing 37: RewardDistributor.sol

```
997  function _swapAndSendBNB(address recipient, uint256 amount) private {
998      IUniswapV2Router02 pcsV2Router = IUniswapV2Router02(router);
999      address[] memory path = new address[](2);
1000     path[0] = rewardToken;
1001     path[1] = pcsV2Router.WETH();
1002     IERC20(rewardToken).approve(address(pcsV2Router), amount);
1003     pcsV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(
1004         amount,
1005         0,
1006         path,
1007         recipient,
1008         block.timestamp.add(30)
1009     );
1010 }
```

Listing 38: RewardDistributor.sol

```
1015 function _swapAndSendToken(address recipient, uint256 amount, address
     ↪ token ) private {
1016     IUniswapV2Router02 pcsV2Router = IUniswapV2Router02(router);
1017     address[] memory path = new address[](3);
1018     path[0] = rewardToken;
1019     path[1] = pcsV2Router.WETH();
1020     path[2] = token;
1021     IERC20(rewardToken).approve(address(pcsV2Router), amount);
1022     pcsV2Router.swapExactTokensForTokensSupportingFeeOnTransferTokens(
1023         amount,
1024         0,
1025         path,
1026         recipient,
```

```
1027            block.timestamp.add(30)
1028        );
1029    }
```

Risk Level:

Likelihood – 2
Impact - 2

Recommendation:

You can use an Oracle to get the exact time or verify if a delay of 900 seconds won't destroy the logic of the staking contract.

Status - Acknowledged

The team has acknowledged the risk.

B.4 Missing Address Verification [LOW]

Description:

Code:

Listing 39: RewardDistributor.sol

```
733 function transferTokenOwnership(address newOwner) external onlyOwner {
734     tokenOwner = newOwner;
735     emit TransferedTokenOwnership(newOwner);
736 }
```

Listing 40: RewardDistributor.sol

```
750 function addRewardHolderShare(address rewardRecipient, uint256 amount)
        ↪ external override onlyToken {
751     referralShares[rewardRecipient].amount = referralShares[
            ↪ rewardRecipient]
```

```
752             .amount
753             .add(amount);
754         totalReferralShares = totalReferralShares.add(amount);
755         rewardRecipientIndexes[rewardRecipient] = rewardRecipients.length;
756         rewardRecipients.push(rewardRecipient);
757         emit AddRewardHolderShare(rewardRecipient, amount);
758     }
```

Listing 41: RewardDistributor.sol

```
766     function removeRewardHolderShare(address rewardRecipient) external
        ↪ onlyOwner {
767         totalReferralShares = totalReferralShares.sub(
768             referralShares[rewardRecipient].amount
769         );
770         referralShares[rewardRecipient].amount = 0;
771         rewardRecipients[
772         rewardRecipientIndexes[rewardRecipient]
```

Risk Level:

Likelihood – 2

Impact - 2

Recommendation:

It is recommended to undertake further validation prior to user-supplied data. The concerns can be resolved by utilizing a whitelist technique or a modifier.

Listing 42: RewardDistributor.sol

```
733     function transferTokenOwnership(address newOwner) external onlyOwner {
734         require(newOwner!= address(0), "Invalid newOwner!");
735         tokenOwner = newOwner;
736         emit TransferedTokenOwnership(newOwner);
737     }
```

Status - Acknowledged

The team has acknowledged the risk.

4 Static Analysis (Slither)

Description:

ShellBoxes expanded the coverage of the specific contract areas using automated testing methodologies. Slither, a Solidity static analysis framework, was one of the tools used. Slither was run on all-scoped contracts in both text and binary formats. This tool can be used to test mathematical relationships between Solidity instances statically and variables that allow for the detection of errors or inconsistent usage of the contracts' APIs throughout the entire codebase.

Results:

```
// HuhToken-30-10-21.sol
Compilation warnings/errors on Desktop/Huh/HuhToken-30-10-21.sol:
Warning: Contract code size exceeds 24576 bytes (a limit introduced in
    ↪ Spurious Dragon). This contract may not be deployable on mainnet.
    ↪  Consider enabling the optimizer (with a low "runs" value!),
    ↪ turning off revert strings, or using libraries.
   --> Desktop/Huh/HuhToken-30-10-21.sol:613:1:
    |
613 | contract HuhToken is Context, IBEP20, Ownable {
    | ^ (Relevant source part starts here and spans across multiple
       ↪ lines).



HuhToken._swapAndAddToLiquidity() (Desktop/Huh/HuhToken-30-10-21.sol
    ↪ #1212-1243) sends eth to arbitrary user
        Dangerous calls:
        - pcsV2Router.addLiquidityETH{value: differenceBnb}(address(this)
            ↪ ,amountAnotherHalf,0,0,_DEAD_ADDRESS,block.timestamp.add
            ↪ (30)) (Desktop/Huh/HuhToken-30-10-21.sol#1233-1240)
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #functions-that-send-ether-to-arbitrary-destinations

Reentrancy in HuhToken._transfer(address,address,uint256) (Desktop/Huh/
    ↪ HuhToken-30-10-21.sol#1046-1088):
        External calls:
        - _swapAndAddToLiquidity() (Desktop/Huh/HuhToken-30-10-21.sol
            ↪ #1062)
                - pcsV2Router.
                    ↪ swapExactTokensForETHSupportingFeeOnTransferTokens(
                    ↪ amountToSwap,0,path,address(this),block.timestamp.
                    ↪ add(30)) (Desktop/Huh/HuhToken-30-10-21.sol
                    ↪ #1223-1229)
                - pcsV2Router.addLiquidityETH{value: differenceBnb}(
                    ↪ address(this),amountAnotherHalf,0,0,_DEAD_ADDRESS,
                    ↪ block.timestamp.add(30)) (Desktop/Huh/HuhToken
                    ↪ -30-10-21.sol#1233-1240)
        External calls sending eth:
        - _swapAndAddToLiquidity() (Desktop/Huh/HuhToken-30-10-21.sol
            ↪ #1062)
                - pcsV2Router.addLiquidityETH{value: differenceBnb}(
                    ↪ address(this),amountAnotherHalf,0,0,_DEAD_ADDRESS,
                    ↪ block.timestamp.add(30)) (Desktop/Huh/HuhToken
                    ↪ -30-10-21.sol#1233-1240)
        State variables written after the call(s):
        - _basicTransfer(sender,recipient,amount) (Desktop/Huh/HuhToken
            ↪ -30-10-21.sol#1065)
                - _rOwned[sender] = _rOwned[sender].sub(rAmount) (Desktop/
                    ↪ Huh/HuhToken-30-10-21.sol#1092)
                - _rOwned[recipient] = _rOwned[recipient].add(rAmount) (
                    ↪ Desktop/Huh/HuhToken-30-10-21.sol#1093)
        - _whitelistedSell(sender,recipient,amount) (Desktop/Huh/HuhToken
            ↪ -30-10-21.sol#1069)
```

```
        - _rOwned[marketingWallet] = _rOwned[marketingWallet].add(
            ↪ rMarketingFee) (Desktop/Huh/HuhToken-30-10-21.sol
            ↪ #1199)
        - _rOwned[sender] = _rOwned[sender].sub(rAmount) (Desktop/
            ↪ Huh/HuhToken-30-10-21.sol#1174)
        - _rOwned[recipient] = _rOwned[recipient].add(
            ↪ rTransferAmount) (Desktop/Huh/HuhToken-30-10-21.sol
            ↪ #1175)
        - _rOwned[address(this)] = _rOwned[address(this)].add(
            ↪ rLiquidityFee) (Desktop/Huh/HuhToken-30-10-21.sol
            ↪ #1176)
    - _normalSell(sender,recipient,amount) (Desktop/Huh/HuhToken
        ↪ -30-10-21.sol#1071)
        - _rOwned[marketingWallet] = _rOwned[marketingWallet].add(
            ↪ rMarketingFee) (Desktop/Huh/HuhToken-30-10-21.sol
            ↪ #1199)
        - _rOwned[sender] = _rOwned[sender].sub(rAmount) (Desktop/
            ↪ Huh/HuhToken-30-10-21.sol#1153)
        - _rOwned[recipient] = _rOwned[recipient].add(
            ↪ rTransferAmount) (Desktop/Huh/HuhToken-30-10-21.sol
            ↪ #1154)
        - _rOwned[address(this)] = _rOwned[address(this)].add(
            ↪ rLiquidityFee) (Desktop/Huh/HuhToken-30-10-21.sol
            ↪ #1155)
    - _normalBuy(sender,recipient,amount) (Desktop/Huh/HuhToken
        ↪ -30-10-21.sol#1078)
        - _rOwned[marketingWallet] = _rOwned[marketingWallet].add(
            ↪ rMarketingFee) (Desktop/Huh/HuhToken-30-10-21.sol
            ↪ #1199)
        - _rOwned[sender] = _rOwned[sender].sub(rAmount) (Desktop/
            ↪ Huh/HuhToken-30-10-21.sol#1107)
        - _rOwned[recipient] = _rOwned[recipient].add(
            ↪ rTransferAmount) (Desktop/Huh/HuhToken-30-10-21.sol
            ↪ #1108)
```

```
        - _rOwned[address(this)] = _rOwned[address(this)].add(
            ↪ rLiquidityFee) (Desktop/Huh/HuhToken-30-10-21.sol
            ↪ #1109)
- _basicTransfer(sender,recipient,amount) (Desktop/Huh/HuhToken
    ↪ -30-10-21.sol#1081)
        - _rOwned[sender] = _rOwned[sender].sub(rAmount) (Desktop/
            ↪ Huh/HuhToken-30-10-21.sol#1092)
        - _rOwned[recipient] = _rOwned[recipient].add(rAmount) (
            ↪ Desktop/Huh/HuhToken-30-10-21.sol#1093)
- _whitelistedSell(sender,recipient,amount) (Desktop/Huh/HuhToken
    ↪ -30-10-21.sol#1069)
        - _rTotal = _rTotal.sub(rFee) (Desktop/Huh/HuhToken
            ↪ -30-10-21.sol#1246)
- _normalSell(sender,recipient,amount) (Desktop/Huh/HuhToken
    ↪ -30-10-21.sol#1071)
        - _rTotal = _rTotal.sub(rFee) (Desktop/Huh/HuhToken
            ↪ -30-10-21.sol#1246)
- _normalBuy(sender,recipient,amount) (Desktop/Huh/HuhToken
    ↪ -30-10-21.sol#1078)
        - _rTotal = _rTotal.sub(rFee) (Desktop/Huh/HuhToken
            ↪ -30-10-21.sol#1246)
- _basicTransfer(sender,recipient,amount) (Desktop/Huh/HuhToken
    ↪ -30-10-21.sol#1065)
        - _tOwned[sender] = _tOwned[sender].sub(amount) (Desktop/
            ↪ Huh/HuhToken-30-10-21.sol#1094)
        - _tOwned[recipient] = _tOwned[recipient].add(amount) (
            ↪ Desktop/Huh/HuhToken-30-10-21.sol#1095)
- _whitelistedSell(sender,recipient,amount) (Desktop/Huh/HuhToken
    ↪ -30-10-21.sol#1069)
        - _tOwned[marketingWallet] = _tOwned[marketingWallet].add(
            ↪ tMarketingFee) (Desktop/Huh/HuhToken-30-10-21.sol
            ↪ #1200)
        - _tOwned[sender] = _tOwned[sender].sub(amount) (Desktop/
            ↪ Huh/HuhToken-30-10-21.sol#1177)
```

```
        - _tOwned[recipient] = _tOwned[recipient].add(
            ↪ rTransferAmount.div(currentRate)) (Desktop/Huh/
            ↪ HuhToken-30-10-21.sol#1178)
        - _tOwned[address(this)] = _tOwned[address(this)].add(
            ↪ rLiquidityFee.div(currentRate)) (Desktop/Huh/
            ↪ HuhToken-30-10-21.sol#1179)
    - _normalSell(sender,recipient,amount) (Desktop/Huh/HuhToken
        ↪ -30-10-21.sol#1071)
        - _tOwned[marketingWallet] = _tOwned[marketingWallet].add(
            ↪ tMarketingFee) (Desktop/Huh/HuhToken-30-10-21.sol
            ↪ #1200)
        - _tOwned[sender] = _tOwned[sender].sub(amount) (Desktop/
            ↪ Huh/HuhToken-30-10-21.sol#1156)
        - _tOwned[recipient] = _tOwned[recipient].add(
            ↪ rTransferAmount.div(currentRate)) (Desktop/Huh/
            ↪ HuhToken-30-10-21.sol#1157)
        - _tOwned[address(this)] = _tOwned[address(this)].add(
            ↪ rLiquidityFee.div(currentRate)) (Desktop/Huh/
            ↪ HuhToken-30-10-21.sol#1158)
    - _normalBuy(sender,recipient,amount) (Desktop/Huh/HuhToken
        ↪ -30-10-21.sol#1078)
        - _tOwned[marketingWallet] = _tOwned[marketingWallet].add(
            ↪ tMarketingFee) (Desktop/Huh/HuhToken-30-10-21.sol
            ↪ #1200)
        - _tOwned[sender] = _tOwned[sender].sub(amount) (Desktop/
            ↪ Huh/HuhToken-30-10-21.sol#1110)
        - _tOwned[recipient] = _tOwned[recipient].add(
            ↪ rTransferAmount.div(currentRate)) (Desktop/Huh/
            ↪ HuhToken-30-10-21.sol#1111)
        - _tOwned[address(this)] = _tOwned[address(this)].add(
            ↪ rLiquidityFee.div(currentRate)) (Desktop/Huh/
            ↪ HuhToken-30-10-21.sol#1112)
    - _basicTransfer(sender,recipient,amount) (Desktop/Huh/HuhToken
        ↪ -30-10-21.sol#1081)
```

```
            - _tOwned[sender] = _tOwned[sender].sub(amount) (Desktop/
                ↪ Huh/HuhToken-30-10-21.sol#1094)
            - _tOwned[recipient] = _tOwned[recipient].add(amount) (
                ↪ Desktop/Huh/HuhToken-30-10-21.sol#1095)
Reentrancy in HuhToken._transfer(address,address,uint256) (Desktop/Huh/
    ↪ HuhToken-30-10-21.sol#1046-1088):
        External calls:
        - _swapAndAddToLiquidity() (Desktop/Huh/HuhToken-30-10-21.sol
            ↪ #1062)
            - pcsV2Router.
                ↪ swapExactTokensForETHSupportingFeeOnTransferTokens(
                ↪ amountToSwap,0,path,address(this),block.timestamp.
                ↪ add(30)) (Desktop/Huh/HuhToken-30-10-21.sol
                ↪ #1223-1229)
            - pcsV2Router.addLiquidityETH{value: differenceBnb}(
                ↪ address(this),amountAnotherHalf,0,0,_DEAD_ADDRESS,
                ↪ block.timestamp.add(30)) (Desktop/Huh/HuhToken
                ↪ -30-10-21.sol#1233-1240)
        - _whitelistedBuy(sender,recipient,amount) (Desktop/Huh/HuhToken
            ↪ -30-10-21.sol#1075)
            - rewardDistributor.addRewardHolderShare(rewardRecipient,
                ↪ tAmount) (Desktop/Huh/HuhToken-30-10-21.sol#1193)
        External calls sending eth:
        - _swapAndAddToLiquidity() (Desktop/Huh/HuhToken-30-10-21.sol
            ↪ #1062)
            - pcsV2Router.addLiquidityETH{value: differenceBnb}(
                ↪ address(this),amountAnotherHalf,0,0,_DEAD_ADDRESS,
                ↪ block.timestamp.add(30)) (Desktop/Huh/HuhToken
                ↪ -30-10-21.sol#1233-1240)
        State variables written after the call(s):
        - _whitelistedBuy(sender,recipient,amount) (Desktop/Huh/HuhToken
            ↪ -30-10-21.sol#1075)
            - _rOwned[address(rewardDistributor)] = _rOwned[address(
                ↪ rewardDistributor)].add(rAmount) (Desktop/Huh/
```

```
            ↪ HuhToken-30-10-21.sol#1189)
        - _rOwned[marketingWallet] = _rOwned[marketingWallet].add(
            ↪ rMarketingFee) (Desktop/Huh/HuhToken-30-10-21.sol
            ↪ #1199)
        - _rOwned[sender] = _rOwned[sender].sub(rAmount) (Desktop/
            ↪ Huh/HuhToken-30-10-21.sol#1130)
        - _rOwned[recipient] = _rOwned[recipient].add(
            ↪ rTransferAmount) (Desktop/Huh/HuhToken-30-10-21.sol
            ↪ #1131)
        - _rOwned[address(this)] = _rOwned[address(this)].add(
            ↪ rLiquidityFee) (Desktop/Huh/HuhToken-30-10-21.sol
            ↪ #1132)
- _whitelistedBuy(sender,recipient,amount) (Desktop/Huh/HuhToken
    ↪ -30-10-21.sol#1075)
        - _rTotal = _rTotal.sub(rFee) (Desktop/Huh/HuhToken
            ↪ -30-10-21.sol#1246)
- _whitelistedBuy(sender,recipient,amount) (Desktop/Huh/HuhToken
    ↪ -30-10-21.sol#1075)
        - _tOwned[marketingWallet] = _tOwned[marketingWallet].add(
            ↪ tMarketingFee) (Desktop/Huh/HuhToken-30-10-21.sol
            ↪ #1200)
        - _tOwned[address(rewardDistributor)] = _tOwned[address(
            ↪ rewardDistributor)].add(tAmount) (Desktop/Huh/
            ↪ HuhToken-30-10-21.sol#1190)
        - _tOwned[sender] = _tOwned[sender].sub(amount) (Desktop/
            ↪ Huh/HuhToken-30-10-21.sol#1134)
        - _tOwned[recipient] = _tOwned[recipient].add(
            ↪ rTransferAmount.div(currentRate)) (Desktop/Huh/
            ↪ HuhToken-30-10-21.sol#1135)
        - _tOwned[address(this)] = _tOwned[address(this)].add(
            ↪ rLiquidityFee.div(currentRate)) (Desktop/Huh/
            ↪ HuhToken-30-10-21.sol#1136)
- isFirstBuy[recipient] = false (Desktop/Huh/HuhToken-30-10-21.
    ↪ sol#1076)
```

```
        - launchedAt = block.number (Desktop/Huh/HuhToken-30-10-21.sol
            ↪ #1086)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #reentrancy-vulnerabilities

HuhToken.withdrawTokens(address,address) (Desktop/Huh/HuhToken-30-10-21.
    ↪ sol#757-765) ignores return value by IBEP20(token).transfer(
    ↪ recipient,balance) (Desktop/Huh/HuhToken-30-10-21.sol#763)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #unchecked-transfer

HuhToken._normalBuy(address,address,uint256) (Desktop/Huh/HuhToken
    ↪ -30-10-21.sol#1100-1119) performs a multiplication on the result
    ↪ of a division:
        -rLiquidityFee = amount.div(100).mul(liquidityFeeOnBuy).mul(
            ↪ currentRate) (Desktop/Huh/HuhToken-30-10-21.sol#1103)
HuhToken._normalBuy(address,address,uint256) (Desktop/Huh/HuhToken
    ↪ -30-10-21.sol#1100-1119) performs a multiplication on the result
    ↪ of a division:
        -rHuhdistributionFee = amount.div(100).mul(
            ↪ HuHdistributionFeeOnBuy).mul(currentRate) (Desktop/Huh/
            ↪ HuhToken-30-10-21.sol#1104)
HuhToken._normalBuy(address,address,uint256) (Desktop/Huh/HuhToken
    ↪ -30-10-21.sol#1100-1119) performs a multiplication on the result
    ↪ of a division:
        -rMarketingFee = amount.div(100).mul(marketingFeeOnBuy).mul(
            ↪ currentRate) (Desktop/Huh/HuhToken-30-10-21.sol#1105)
HuhToken._whitelistedBuy(address,address,uint256) (Desktop/Huh/HuhToken
    ↪ -30-10-21.sol#1121-1144) performs a multiplication on the result
    ↪ of a division:
        -tReferralRewardAmount = amount.div(100).mul(referralReward) (
            ↪ Desktop/Huh/HuhToken-30-10-21.sol#1124)
HuhToken._whitelistedBuy(address,address,uint256) (Desktop/Huh/HuhToken
    ↪ -30-10-21.sol#1121-1144) performs a multiplication on the result
```

```
    ↪ of a division:
        -rLiquidityFee = amount.div(100).mul(liquidityFeeOnWhiteListedBuy
           ↪ ).mul(currentRate) (Desktop/Huh/HuhToken-30-10-21.sol
           ↪ #1126)
HuhToken._whitelistedBuy(address,address,uint256) (Desktop/Huh/HuhToken
    ↪ -30-10-21.sol#1121-1144) performs a multiplication on the result
    ↪ of a division:
        -rHuhdistributionFee = amount.div(100).mul(
           ↪ HuHdistributionFeeOnBuyWhiteListed).mul(currentRate) (
           ↪ Desktop/Huh/HuhToken-30-10-21.sol#1127)
HuhToken._whitelistedBuy(address,address,uint256) (Desktop/Huh/HuhToken
    ↪ -30-10-21.sol#1121-1144) performs a multiplication on the result
    ↪ of a division:
        -rMarketingFee = amount.div(100).mul(marketingFeeOnBuyWhiteListed
           ↪ ).mul(currentRate) (Desktop/Huh/HuhToken-30-10-21.sol
           ↪ #1128)
HuhToken._normalSell(address,address,uint256) (Desktop/Huh/HuhToken
    ↪ -30-10-21.sol#1146-1165) performs a multiplication on the result
    ↪ of a division:
        -rLiquidityFee = amount.div(100).mul(liquidityFeeOnSell).mul(
           ↪ currentRate) (Desktop/Huh/HuhToken-30-10-21.sol#1149)
HuhToken._normalSell(address,address,uint256) (Desktop/Huh/HuhToken
    ↪ -30-10-21.sol#1146-1165) performs a multiplication on the result
    ↪ of a division:
        -rHuhdistributionFee = amount.div(100).mul(
           ↪ HuHdistributionFeeOnSell).mul(currentRate) (Desktop/Huh/
           ↪ HuhToken-30-10-21.sol#1150)
HuhToken._normalSell(address,address,uint256) (Desktop/Huh/HuhToken
    ↪ -30-10-21.sol#1146-1165) performs a multiplication on the result
    ↪ of a division:
        -rMarketingFee = amount.div(100).mul(marketingFeeOnSell).mul(
           ↪ currentRate) (Desktop/Huh/HuhToken-30-10-21.sol#1151)
HuhToken._whitelistedSell(address,address,uint256) (Desktop/Huh/HuhToken
    ↪ -30-10-21.sol#1167-1186) performs a multiplication on the result
```

```
    ↪ of a division:
        -rLiquidityFee = amount.div(100).mul(
            ↪ liquidityFeeOnWhiteListedSell).mul(currentRate) (Desktop/
            ↪ Huh/HuhToken-30-10-21.sol#1170)
HuhToken._whitelistedSell(address,address,uint256) (Desktop/Huh/HuhToken
    ↪ -30-10-21.sol#1167-1186) performs a multiplication on the result
    ↪ of a division:
        -rHuhdistributionFee = amount.div(100).mul(
            ↪ HuHdistributionFeeOnWhiteListedSell).mul(currentRate) (
            ↪ Desktop/Huh/HuhToken-30-10-21.sol#1171)
HuhToken._whitelistedSell(address,address,uint256) (Desktop/Huh/HuhToken
    ↪ -30-10-21.sol#1167-1186) performs a multiplication on the result
    ↪ of a division:
        -rMarketingFee = amount.div(100).mul(
            ↪ marketingFeeOnWhiteListedSell).mul(currentRate) (Desktop/
            ↪ Huh/HuhToken-30-10-21.sol#1172)
HuhToken.slitherConstructorVariables() (Desktop/Huh/HuhToken-30-10-21.
    ↪ sol#613-1292) performs a multiplication on the result of a
    ↪ division:
        -maxTxAmount = _tTotal.mul(1).div(10 ** 2) (Desktop/Huh/HuhToken
            ↪ -30-10-21.sol#686)
        -amountOfTokensToAddToLiquidityThreshold = maxTxAmount.mul(10).
            ↪ div(10 ** 2) (Desktop/Huh/HuhToken-30-10-21.sol#687)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #divide-before-multiply

Reentrancy in HuhToken._whitelistedBuy(address,address,uint256) (Desktop
    ↪ /Huh/HuhToken-30-10-21.sol#1121-1144):
        External calls:
        - _sendToRewardDistributor(sender,referParent[recipient],
            ↪ tReferralRewardAmount,rReferralRewardAmount) (Desktop/Huh/
            ↪ HuhToken-30-10-21.sol#1141)
                - rewardDistributor.addRewardHolderShare(rewardRecipient,
                    ↪ tAmount) (Desktop/Huh/HuhToken-30-10-21.sol#1193)
```

```
        State variables written after the call(s):
        - _sendToMarketingWallet(sender,rMarketingFee.div(currentRate),
            ↪ rMarketingFee) (Desktop/Huh/HuhToken-30-10-21.sol#1142)
                - _rOwned[marketingWallet] = _rOwned[marketingWallet].add(
                    ↪ rMarketingFee) (Desktop/Huh/HuhToken-30-10-21.sol
                    ↪ #1199)
        - _reflectFee(rHuhdistributionFee,rHuhdistributionFee.div(
            ↪ currentRate)) (Desktop/Huh/HuhToken-30-10-21.sol#1143)
                - _rTotal = _rTotal.sub(rFee) (Desktop/Huh/HuhToken
                    ↪ -30-10-21.sol#1246)
        - _sendToMarketingWallet(sender,rMarketingFee.div(currentRate),
            ↪ rMarketingFee) (Desktop/Huh/HuhToken-30-10-21.sol#1142)
                - _tOwned[marketingWallet] = _tOwned[marketingWallet].add(
                    ↪ tMarketingFee) (Desktop/Huh/HuhToken-30-10-21.sol
                    ↪ #1200)
    Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
        ↪ #reentrancy-vulnerabilities-1

    HuhToken._swapAndAddToLiquidity() (Desktop/Huh/HuhToken-30-10-21.sol
        ↪ #1212-1243) ignores return value by pcsV2Router.addLiquidityETH{
        ↪ value: differenceBnb}(address(this),amountAnotherHalf,0,0,
        ↪ _DEAD_ADDRESS,block.timestamp.add(30)) (Desktop/Huh/HuhToken
        ↪ -30-10-21.sol#1233-1240)
    Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
        ↪ #unused-return

    HuhToken.allowance(address,address).owner (Desktop/Huh/HuhToken
        ↪ -30-10-21.sol#964) shadows:
            - Ownable.owner() (Desktop/Huh/HuhToken-30-10-21.sol#53-55) (
                ↪ function)
    HuhToken._approve(address,address,uint256).owner (Desktop/Huh/HuhToken
        ↪ -30-10-21.sol#1038) shadows:
            - Ownable.owner() (Desktop/Huh/HuhToken-30-10-21.sol#53-55) (
                ↪ function)
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #local-variable-shadowing

Reentrancy in HuhToken._sendToRewardDistributor(address,address,uint256,
    ↪ uint256) (Desktop/Huh/HuhToken-30-10-21.sol#1188-1196):
        External calls:
        - rewardDistributor.addRewardHolderShare(rewardRecipient,tAmount)
            ↪  (Desktop/Huh/HuhToken-30-10-21.sol#1193)
        State variables written after the call(s):
        - totalReferralReward = totalReferralReward.add(tAmount) (Desktop
            ↪ /Huh/HuhToken-30-10-21.sol#1195)
        - userReferralReward[rewardRecipient] = userReferralReward[
            ↪ rewardRecipient].add(tAmount) (Desktop/Huh/HuhToken
            ↪ -30-10-21.sol#1194)
Reentrancy in HuhToken._transfer(address,address,uint256) (Desktop/Huh/
    ↪ HuhToken-30-10-21.sol#1046-1088):
        External calls:
        - _swapAndAddToLiquidity() (Desktop/Huh/HuhToken-30-10-21.sol
            ↪ #1062)
                - pcsV2Router.
                    ↪ swapExactTokensForETHSupportingFeeOnTransferTokens(
                    ↪ amountToSwap,0,path,address(this),block.timestamp.
                    ↪ add(30)) (Desktop/Huh/HuhToken-30-10-21.sol
                    ↪ #1223-1229)
                - pcsV2Router.addLiquidityETH{value: differenceBnb}(
                    ↪ address(this),amountAnotherHalf,0,0,_DEAD_ADDRESS,
                    ↪ block.timestamp.add(30)) (Desktop/Huh/HuhToken
                    ↪ -30-10-21.sol#1233-1240)
        External calls sending eth:
        - _swapAndAddToLiquidity() (Desktop/Huh/HuhToken-30-10-21.sol
            ↪ #1062)
                - pcsV2Router.addLiquidityETH{value: differenceBnb}(
                    ↪ address(this),amountAnotherHalf,0,0,_DEAD_ADDRESS,
                    ↪ block.timestamp.add(30)) (Desktop/Huh/HuhToken
```

```
                ↪ -30-10-21.sol#1233-1240)
        State variables written after the call(s):
        - _whitelistedSell(sender,recipient,amount) (Desktop/Huh/HuhToken
            ↪ -30-10-21.sol#1069)
                - _tFeeTotal = _tFeeTotal.add(tFee) (Desktop/Huh/HuhToken
                    ↪ -30-10-21.sol#1247)
        - _normalSell(sender,recipient,amount) (Desktop/Huh/HuhToken
            ↪ -30-10-21.sol#1071)
                - _tFeeTotal = _tFeeTotal.add(tFee) (Desktop/Huh/HuhToken
                    ↪ -30-10-21.sol#1247)
        - _normalBuy(sender,recipient,amount) (Desktop/Huh/HuhToken
            ↪ -30-10-21.sol#1078)
                - _tFeeTotal = _tFeeTotal.add(tFee) (Desktop/Huh/HuhToken
                    ↪ -30-10-21.sol#1247)
    Reentrancy in HuhToken._whitelistedBuy(address,address,uint256) (Desktop
        ↪ /Huh/HuhToken-30-10-21.sol#1121-1144):
        External calls:
        - _sendToRewardDistributor(sender,referParent[recipient],
            ↪ tReferralRewardAmount,rReferralRewardAmount) (Desktop/Huh/
            ↪ HuhToken-30-10-21.sol#1141)
                - rewardDistributor.addRewardHolderShare(rewardRecipient,
                    ↪ tAmount) (Desktop/Huh/HuhToken-30-10-21.sol#1193)
        State variables written after the call(s):
        - _reflectFee(rHuhdistributionFee,rHuhdistributionFee.div(
            ↪ currentRate)) (Desktop/Huh/HuhToken-30-10-21.sol#1143)
                - _tFeeTotal = _tFeeTotal.add(tFee) (Desktop/Huh/HuhToken
                    ↪ -30-10-21.sol#1247)
    Reentrancy in HuhToken.transferFrom(address,address,uint256) (Desktop/
        ↪ Huh/HuhToken-30-10-21.sol#917-924):
        External calls:
        - _transfer(sender,recipient,amount) (Desktop/Huh/HuhToken
            ↪ -30-10-21.sol#918)
                - rewardDistributor.addRewardHolderShare(rewardRecipient,
                    ↪ tAmount) (Desktop/Huh/HuhToken-30-10-21.sol#1193)
```

```
            - pcsV2Router.
                ↪ swapExactTokensForETHSupportingFeeOnTransferTokens(
                ↪ amountToSwap,0,path,address(this),block.timestamp.
                ↪ add(30)) (Desktop/Huh/HuhToken-30-10-21.sol
                ↪ #1223-1229)
            - pcsV2Router.addLiquidityETH{value: differenceBnb}(
                ↪ address(this),amountAnotherHalf,0,0,_DEAD_ADDRESS,
                ↪ block.timestamp.add(30)) (Desktop/Huh/HuhToken
                ↪ -30-10-21.sol#1233-1240)
        External calls sending eth:
        - _transfer(sender,recipient,amount) (Desktop/Huh/HuhToken
            ↪ -30-10-21.sol#918)
            - pcsV2Router.addLiquidityETH{value: differenceBnb}(
                ↪ address(this),amountAnotherHalf,0,0,_DEAD_ADDRESS,
                ↪ block.timestamp.add(30)) (Desktop/Huh/HuhToken
                ↪ -30-10-21.sol#1233-1240)
        State variables written after the call(s):
        - _approve(sender,_msgSender(),_allowances[sender][_msgSender()].
            ↪ sub(amount,BEP20: transfer amount exceeds allowance)) (
            ↪ Desktop/Huh/HuhToken-30-10-21.sol#919-922)
            - _allowances[owner][spender] = amount (Desktop/Huh/
                ↪ HuhToken-30-10-21.sol#1042)
Reentrancy in HuhToken.updatePancakeSwapRouter(address) (Desktop/Huh/
    ↪ HuhToken-30-10-21.sol#835-843):
        External calls:
        - pcsV2Pair = IUniswapV2Factory(pcsV2Router.factory()).createPair
            ↪ (address(this),pcsV2Router.WETH()) (Desktop/Huh/HuhToken
            ↪ -30-10-21.sol#839)
        State variables written after the call(s):
        - _allowances[address(this)][address(pcsV2Router)] = _MAX (
            ↪ Desktop/Huh/HuhToken-30-10-21.sol#840)
        - _allowances[address(rewardDistributor)][address(pcsV2Router)] =
            ↪  _MAX (Desktop/Huh/HuhToken-30-10-21.sol#841)
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #reentrancy-vulnerabilities-2

Reentrancy in HuhToken._swapAndAddToLiquidity() (Desktop/Huh/HuhToken
    ↪ -30-10-21.sol#1212-1243):
        External calls:
        - pcsV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(
            ↪ amountToSwap,0,path,address(this),block.timestamp.add(30))
            ↪  (Desktop/Huh/HuhToken-30-10-21.sol#1223-1229)
        - pcsV2Router.addLiquidityETH{value: differenceBnb}(address(this)
            ↪ ,amountAnotherHalf,0,0,_DEAD_ADDRESS,block.timestamp.add
            ↪ (30)) (Desktop/Huh/HuhToken-30-10-21.sol#1233-1240)
        External calls sending eth:
        - pcsV2Router.addLiquidityETH{value: differenceBnb}(address(this)
            ↪ ,amountAnotherHalf,0,0,_DEAD_ADDRESS,block.timestamp.add
            ↪ (30)) (Desktop/Huh/HuhToken-30-10-21.sol#1233-1240)
        Event emitted after the call(s):
        - SwapAndLiquify(differenceBnb,amountToSwap) (Desktop/Huh/
            ↪ HuhToken-30-10-21.sol#1242)
Reentrancy in HuhToken._transfer(address,address,uint256) (Desktop/Huh/
    ↪ HuhToken-30-10-21.sol#1046-1088):
        External calls:
        - _swapAndAddToLiquidity() (Desktop/Huh/HuhToken-30-10-21.sol
            ↪ #1062)
                - pcsV2Router.
                    ↪ swapExactTokensForETHSupportingFeeOnTransferTokens(
                    ↪ amountToSwap,0,path,address(this),block.timestamp.
                    ↪ add(30)) (Desktop/Huh/HuhToken-30-10-21.sol
                    ↪ #1223-1229)
                - pcsV2Router.addLiquidityETH{value: differenceBnb}(
                    ↪ address(this),amountAnotherHalf,0,0,_DEAD_ADDRESS,
                    ↪ block.timestamp.add(30)) (Desktop/Huh/HuhToken
                    ↪ -30-10-21.sol#1233-1240)
        External calls sending eth:
```

```
- _swapAndAddToLiquidity() (Desktop/Huh/HuhToken-30-10-21.sol
    ↪ #1062)
        - pcsV2Router.addLiquidityETH{value: differenceBnb}(
            ↪ address(this),amountAnotherHalf,0,0,_DEAD_ADDRESS,
            ↪ block.timestamp.add(30)) (Desktop/Huh/HuhToken
            ↪ -30-10-21.sol#1233-1240)
Event emitted after the call(s):
- Transfer(sender,marketingWallet,tMarketingFee) (Desktop/Huh/
    ↪ HuhToken-30-10-21.sol#1201)
        - _whitelistedSell(sender,recipient,amount) (Desktop/Huh/
            ↪ HuhToken-30-10-21.sol#1069)
- Transfer(sender,marketingWallet,tMarketingFee) (Desktop/Huh/
    ↪ HuhToken-30-10-21.sol#1201)
        - _normalBuy(sender,recipient,amount) (Desktop/Huh/
            ↪ HuhToken-30-10-21.sol#1078)
- Transfer(sender,marketingWallet,tMarketingFee) (Desktop/Huh/
    ↪ HuhToken-30-10-21.sol#1201)
        - _normalSell(sender,recipient,amount) (Desktop/Huh/
            ↪ HuhToken-30-10-21.sol#1071)
- Transfer(sender,recipient,amount) (Desktop/Huh/HuhToken
    ↪ -30-10-21.sol#1097)
        - _basicTransfer(sender,recipient,amount) (Desktop/Huh/
            ↪ HuhToken-30-10-21.sol#1065)
- Transfer(sender,recipient,amount) (Desktop/Huh/HuhToken
    ↪ -30-10-21.sol#1097)
        - _basicTransfer(sender,recipient,amount) (Desktop/Huh/
            ↪ HuhToken-30-10-21.sol#1081)
- Transfer(sender,recipient,rTransferAmount.div(currentRate)) (
    ↪ Desktop/Huh/HuhToken-30-10-21.sol#1181)
        - _whitelistedSell(sender,recipient,amount) (Desktop/Huh/
            ↪ HuhToken-30-10-21.sol#1069)
- Transfer(sender,recipient,rTransferAmount.div(currentRate)) (
    ↪ Desktop/Huh/HuhToken-30-10-21.sol#1114)
```

```
            - _normalBuy(sender,recipient,amount) (Desktop/Huh/
                ↪ HuhToken-30-10-21.sol#1078)
        - Transfer(sender,recipient,rTransferAmount.div(currentRate)) (
            ↪ Desktop/Huh/HuhToken-30-10-21.sol#1160)
            - _normalSell(sender,recipient,amount) (Desktop/Huh/
                ↪ HuhToken-30-10-21.sol#1071)
        - Transfer(sender,address(this),rLiquidityFee.div(currentRate)) (
            ↪ Desktop/Huh/HuhToken-30-10-21.sol#1182)
            - _whitelistedSell(sender,recipient,amount) (Desktop/Huh/
                ↪ HuhToken-30-10-21.sol#1069)
        - Transfer(sender,address(this),rLiquidityFee.div(currentRate)) (
            ↪ Desktop/Huh/HuhToken-30-10-21.sol#1161)
            - _normalSell(sender,recipient,amount) (Desktop/Huh/
                ↪ HuhToken-30-10-21.sol#1071)
        - Transfer(sender,address(this),(rLiquidityFee).div(currentRate))
            ↪  (Desktop/Huh/HuhToken-30-10-21.sol#1115)
            - _normalBuy(sender,recipient,amount) (Desktop/Huh/
                ↪ HuhToken-30-10-21.sol#1078)
    Reentrancy in HuhToken._transfer(address,address,uint256) (Desktop/Huh/
        ↪ HuhToken-30-10-21.sol#1046-1088):
        External calls:
        - _swapAndAddToLiquidity() (Desktop/Huh/HuhToken-30-10-21.sol
            ↪ #1062)
            - pcsV2Router.
                ↪ swapExactTokensForETHSupportingFeeOnTransferTokens(
                ↪ amountToSwap,0,path,address(this),block.timestamp.
                ↪ add(30)) (Desktop/Huh/HuhToken-30-10-21.sol
                ↪ #1223-1229)
            - pcsV2Router.addLiquidityETH{value: differenceBnb}(
                ↪ address(this),amountAnotherHalf,0,0,_DEAD_ADDRESS,
                ↪ block.timestamp.add(30)) (Desktop/Huh/HuhToken
                ↪ -30-10-21.sol#1233-1240)
        - _whitelistedBuy(sender,recipient,amount) (Desktop/Huh/HuhToken
            ↪ -30-10-21.sol#1075)
```

```
            - rewardDistributor.addRewardHolderShare(rewardRecipient,
                ↪ tAmount) (Desktop/Huh/HuhToken-30-10-21.sol#1193)
        External calls sending eth:
        - _swapAndAddToLiquidity() (Desktop/Huh/HuhToken-30-10-21.sol
            ↪ #1062)
            - pcsV2Router.addLiquidityETH{value: differenceBnb}(
                ↪ address(this),amountAnotherHalf,0,0,_DEAD_ADDRESS,
                ↪ block.timestamp.add(30)) (Desktop/Huh/HuhToken
                ↪ -30-10-21.sol#1233-1240)
        Event emitted after the call(s):
        - Transfer(sender,address(rewardDistributor),tAmount) (Desktop/
            ↪ Huh/HuhToken-30-10-21.sol#1192)
            - _whitelistedBuy(sender,recipient,amount) (Desktop/Huh/
                ↪ HuhToken-30-10-21.sol#1075)
        - Transfer(sender,marketingWallet,tMarketingFee) (Desktop/Huh/
            ↪ HuhToken-30-10-21.sol#1201)
            - _whitelistedBuy(sender,recipient,amount) (Desktop/Huh/
                ↪ HuhToken-30-10-21.sol#1075)
        - Transfer(sender,recipient,rTransferAmount.div(currentRate)) (
            ↪ Desktop/Huh/HuhToken-30-10-21.sol#1138)
            - _whitelistedBuy(sender,recipient,amount) (Desktop/Huh/
                ↪ HuhToken-30-10-21.sol#1075)
        - Transfer(sender,address(this),rLiquidityFee.div(currentRate)) (
            ↪ Desktop/Huh/HuhToken-30-10-21.sol#1139)
            - _whitelistedBuy(sender,recipient,amount) (Desktop/Huh/
                ↪ HuhToken-30-10-21.sol#1075)
    Reentrancy in HuhToken._whitelistedBuy(address,address,uint256) (Desktop
        ↪ /Huh/HuhToken-30-10-21.sol#1121-1144):
        External calls:
        - _sendToRewardDistributor(sender,referParent[recipient],
            ↪ tReferralRewardAmount,rReferralRewardAmount) (Desktop/Huh/
            ↪ HuhToken-30-10-21.sol#1141)
            - rewardDistributor.addRewardHolderShare(rewardRecipient,
                ↪ tAmount) (Desktop/Huh/HuhToken-30-10-21.sol#1193)
```

```
        Event emitted after the call(s):
        - Transfer(sender,marketingWallet,tMarketingFee) (Desktop/Huh/
            ↪ HuhToken-30-10-21.sol#1201)
                - _sendToMarketingWallet(sender,rMarketingFee.div(
                    ↪ currentRate),rMarketingFee) (Desktop/Huh/HuhToken
                    ↪ -30-10-21.sol#1142)
    Reentrancy in HuhToken.transferFrom(address,address,uint256) (Desktop/
        ↪ Huh/HuhToken-30-10-21.sol#917-924):
        External calls:
        - _transfer(sender,recipient,amount) (Desktop/Huh/HuhToken
            ↪ -30-10-21.sol#918)
                - rewardDistributor.addRewardHolderShare(rewardRecipient,
                    ↪ tAmount) (Desktop/Huh/HuhToken-30-10-21.sol#1193)
                - pcsV2Router.
                    ↪ swapExactTokensForETHSupportingFeeOnTransferTokens(
                    ↪ amountToSwap,0,path,address(this),block.timestamp.
                    ↪ add(30)) (Desktop/Huh/HuhToken-30-10-21.sol
                    ↪ #1223-1229)
                - pcsV2Router.addLiquidityETH{value: differenceBnb}(
                    ↪ address(this),amountAnotherHalf,0,0,_DEAD_ADDRESS,
                    ↪ block.timestamp.add(30)) (Desktop/Huh/HuhToken
                    ↪ -30-10-21.sol#1233-1240)
        External calls sending eth:
        - _transfer(sender,recipient,amount) (Desktop/Huh/HuhToken
            ↪ -30-10-21.sol#918)
                - pcsV2Router.addLiquidityETH{value: differenceBnb}(
                    ↪ address(this),amountAnotherHalf,0,0,_DEAD_ADDRESS,
                    ↪ block.timestamp.add(30)) (Desktop/Huh/HuhToken
                    ↪ -30-10-21.sol#1233-1240)
        Event emitted after the call(s):
        - Approval(owner,spender,amount) (Desktop/Huh/HuhToken-30-10-21.
            ↪ sol#1043)
                - _approve(sender,_msgSender(),_allowances[sender][
                    ↪ _msgSender()].sub(amount,BEP20: transfer amount
```

```
                ↪ exceeds allowance)) (Desktop/Huh/HuhToken-30-10-21.
                ↪ sol#919-922)
Reentrancy in HuhToken.updatePancakeSwapRouter(address) (Desktop/Huh/
    ↪ HuhToken-30-10-21.sol#835-843):
        External calls:
        - pcsV2Pair = IUniswapV2Factory(pcsV2Router.factory()).createPair
            ↪ (address(this),pcsV2Router.WETH()) (Desktop/Huh/HuhToken
            ↪ -30-10-21.sol#839)
        Event emitted after the call(s):
        - UpdatePancakeSwapRouter(_pcsV2Router) (Desktop/Huh/HuhToken
            ↪ -30-10-21.sol#842)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #reentrancy-vulnerabilities-3

HuhToken._transfer(address,address,uint256) (Desktop/Huh/HuhToken
    ↪ -30-10-21.sol#1046-1088) compares to a boolean constant:
        -require(bool,string)(swapEnabled == true,Swap Is Disabled!) (
            ↪ Desktop/Huh/HuhToken-30-10-21.sol#1051)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #boolean-equality

Context._msgData() (Desktop/Huh/HuhToken-30-10-21.sol#21-23) is never
    ↪ used and should be removed
SafeMath.div(uint256,uint256,string) (Desktop/Huh/HuhToken-30-10-21.sol
    ↪ #277-286) is never used and should be removed
SafeMath.mod(uint256,uint256) (Desktop/Huh/HuhToken-30-10-21.sol
    ↪ #237-239) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (Desktop/Huh/HuhToken-30-10-21.sol
    ↪ #303-312) is never used and should be removed
SafeMath.tryAdd(uint256,uint256) (Desktop/Huh/HuhToken-30-10-21.sol
    ↪ #108-114) is never used and should be removed
SafeMath.tryDiv(uint256,uint256) (Desktop/Huh/HuhToken-30-10-21.sol
    ↪ #150-155) is never used and should be removed
```

```
SafeMath.tryMod(uint256,uint256) (Desktop/Huh/HuhToken-30-10-21.sol
    ↪ #162-167) is never used and should be removed
SafeMath.tryMul(uint256,uint256) (Desktop/Huh/HuhToken-30-10-21.sol
    ↪ #133-143) is never used and should be removed
SafeMath.trySub(uint256,uint256) (Desktop/Huh/HuhToken-30-10-21.sol
    ↪ #121-126) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #dead-code

HuhToken._rTotal (Desktop/Huh/HuhToken-30-10-21.sol#622) is set pre-
    ↪ construction with a non-constant function or state variable:
        - (_MAX - (_MAX % _tTotal))
HuhToken.maxTxAmount (Desktop/Huh/HuhToken-30-10-21.sol#686) is set pre-
    ↪ construction with a non-constant function or state variable:
        - _tTotal.mul(1).div(10 ** 2)
HuhToken.amountOfTokensToAddToLiquidityThreshold (Desktop/Huh/HuhToken
    ↪ -30-10-21.sol#687) is set pre-construction with a non-constant
    ↪ function or state variable:
        - maxTxAmount.mul(10).div(10 ** 2)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #function-initializing-state

Function IUniswapV2Pair.DOMAIN_SEPARATOR() (Desktop/Huh/HuhToken
    ↪ -30-10-21.sol#418) is not in mixedCase
Function IUniswapV2Pair.PERMIT_TYPEHASH() (Desktop/Huh/HuhToken
    ↪ -30-10-21.sol#419) is not in mixedCase
Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (Desktop/Huh/HuhToken
    ↪ -30-10-21.sol#436) is not in mixedCase
Function IUniswapV2Router01.WETH() (Desktop/Huh/HuhToken-30-10-21.sol
    ↪ #456) is not in mixedCase
Parameter HuhToken.changeFeesForNormalBuy(uint8,uint8,uint8).
    ↪ _liquidityFeeOnBuy (Desktop/Huh/HuhToken-30-10-21.sol#785) is not
    ↪  in mixedCase
```

```
Parameter HuhToken.changeFeesForNormalBuy(uint8,uint8,uint8).
    ↪ _marketingFeeOnBuy (Desktop/Huh/HuhToken-30-10-21.sol#785) is not
    ↪  in mixedCase
Parameter HuhToken.changeFeesForNormalBuy(uint8,uint8,uint8).
    ↪ _HuHdistributionFeeOnBuy (Desktop/Huh/HuhToken-30-10-21.sol#785)
    ↪ is not in mixedCase
Parameter HuhToken.changeFeesForWhiteListedBuy(uint8,uint8,uint8).
    ↪ _liquidityFeeOnBuy (Desktop/Huh/HuhToken-30-10-21.sol#792) is not
    ↪  in mixedCase
Parameter HuhToken.changeFeesForWhiteListedBuy(uint8,uint8,uint8).
    ↪ _marketingFeeOnBuy (Desktop/Huh/HuhToken-30-10-21.sol#792) is not
    ↪  in mixedCase
Parameter HuhToken.changeFeesForWhiteListedBuy(uint8,uint8,uint8).
    ↪ _HuHdistributionFeeOnBuy (Desktop/Huh/HuhToken-30-10-21.sol#792)
    ↪ is not in mixedCase
Parameter HuhToken.changeFeesForNormalSell(uint8,uint8,uint8).
    ↪ _liquidityFeeOnSell (Desktop/Huh/HuhToken-30-10-21.sol#799) is
    ↪ not in mixedCase
Parameter HuhToken.changeFeesForNormalSell(uint8,uint8,uint8).
    ↪ _marketingFeeOnSell (Desktop/Huh/HuhToken-30-10-21.sol#799) is
    ↪ not in mixedCase
Parameter HuhToken.changeFeesForNormalSell(uint8,uint8,uint8).
    ↪ _HuHdistributionFeeOnSell (Desktop/Huh/HuhToken-30-10-21.sol#799)
    ↪  is not in mixedCase
Parameter HuhToken.changeFeesForWhitelistedSell(uint8,uint8,uint8).
    ↪ _liquidityFeeOnSell (Desktop/Huh/HuhToken-30-10-21.sol#806) is
    ↪ not in mixedCase
Parameter HuhToken.changeFeesForWhitelistedSell(uint8,uint8,uint8).
    ↪ _marketingFeeOnSell (Desktop/Huh/HuhToken-30-10-21.sol#806) is
    ↪ not in mixedCase
Parameter HuhToken.changeFeesForWhitelistedSell(uint8,uint8,uint8).
    ↪ _HuHdistributionFeeOnSell (Desktop/Huh/HuhToken-30-10-21.sol#806)
    ↪  is not in mixedCase
```

```
Parameter HuhToken.changeReferralReward(uint8)._referralReward (Desktop/
    ↪ Huh/HuhToken-30-10-21.sol#813) is not in mixedCase
Parameter HuhToken.updateMarketingWallet(address)._marketingWallet (
    ↪ Desktop/Huh/HuhToken-30-10-21.sol#818) is not in mixedCase
Parameter HuhToken.setReferralCodeRegistrator(address).
    ↪ _referralCodeRegistrator (Desktop/Huh/HuhToken-30-10-21.sol#824)
    ↪ is not in mixedCase
Parameter HuhToken.updateAmountOfTokensToAddToLiquidityThreshold(uint256
    ↪ )._amountOfTokensToAddToLiquidityThreshold (Desktop/Huh/HuhToken
    ↪ -30-10-21.sol#830) is not in mixedCase
Parameter HuhToken.updatePancakeSwapRouter(address)._pcsV2Router (
    ↪ Desktop/Huh/HuhToken-30-10-21.sol#835) is not in mixedCase
Parameter HuhToken.updateRewardDistributor(address)._rewardDistributor (
    ↪ Desktop/Huh/HuhToken-30-10-21.sol#845) is not in mixedCase
Parameter HuhToken.updateSwapAndLiquifyEnabled(bool).
    ↪ _swapAndLiquifyEnabled (Desktop/Huh/HuhToken-30-10-21.sol#855) is
    ↪  not in mixedCase
Parameter HuhToken.setSwapEnabled(bool)._swapEnabled (Desktop/Huh/
    ↪ HuhToken-30-10-21.sol#861) is not in mixedCase
Variable HuhToken.HuHdistributionFeeOnBuy (Desktop/Huh/HuhToken
    ↪ -30-10-21.sol#639) is not in mixedCase
Variable HuhToken.HuHdistributionFeeOnBuyWhiteListed (Desktop/Huh/
    ↪ HuhToken-30-10-21.sol#643) is not in mixedCase
Variable HuhToken.HuHdistributionFeeOnSell (Desktop/Huh/HuhToken
    ↪ -30-10-21.sol#647) is not in mixedCase
Variable HuhToken.HuHdistributionFeeOnWhiteListedSell (Desktop/Huh/
    ↪ HuhToken-30-10-21.sol#651) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #conformance-to-solidity-naming-conventions

Variable IUniswapV2Router01.addLiquidity(address,address,uint256,uint256
    ↪ ,uint256,uint256,address,uint256).amountADesired (Desktop/Huh/
    ↪ HuhToken-30-10-21.sol#461) is too similar to IUniswapV2Router01.
    ↪ addLiquidity(address,address,uint256,uint256,uint256,uint256,
```

```
    ↪ address,uint256).amountBDesired (Desktop/Huh/HuhToken-30-10-21.
    ↪ sol#462)
Variable HuhToken._whitelistedSell(address,address,uint256).
    ↪ rMarketingFee (Desktop/Huh/HuhToken-30-10-21.sol#1172) is too
    ↪ similar to HuhToken._sendToMarketingWallet(address,uint256,
    ↪ uint256).tMarketingFee (Desktop/Huh/HuhToken-30-10-21.sol#1198)
Variable HuhToken._whitelistedBuy(address,address,uint256).rMarketingFee
    ↪  (Desktop/Huh/HuhToken-30-10-21.sol#1128) is too similar to
    ↪ HuhToken._sendToMarketingWallet(address,uint256,uint256).
    ↪ tMarketingFee (Desktop/Huh/HuhToken-30-10-21.sol#1198)
Variable HuhToken._sendToMarketingWallet(address,uint256,uint256).
    ↪ rMarketingFee (Desktop/Huh/HuhToken-30-10-21.sol#1198) is too
    ↪ similar to HuhToken._sendToMarketingWallet(address,uint256,
    ↪ uint256).tMarketingFee (Desktop/Huh/HuhToken-30-10-21.sol#1198)
Variable HuhToken._normalBuy(address,address,uint256).rMarketingFee (
    ↪ Desktop/Huh/HuhToken-30-10-21.sol#1105) is too similar to
    ↪ HuhToken._sendToMarketingWallet(address,uint256,uint256).
    ↪ tMarketingFee (Desktop/Huh/HuhToken-30-10-21.sol#1198)
Variable HuhToken._normalSell(address,address,uint256).rMarketingFee (
    ↪ Desktop/Huh/HuhToken-30-10-21.sol#1151) is too similar to
    ↪ HuhToken._sendToMarketingWallet(address,uint256,uint256).
    ↪ tMarketingFee (Desktop/Huh/HuhToken-30-10-21.sol#1198)
Variable HuhToken._whitelistedBuy(address,address,uint256).
    ↪ rReferralRewardAmount (Desktop/Huh/HuhToken-30-10-21.sol#1125) is
    ↪  too similar to HuhToken._whitelistedBuy(address,address,uint256)
    ↪ .tReferralRewardAmount (Desktop/Huh/HuhToken-30-10-21.sol#1124)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #variable-names-are-too-similar

HuhToken.slitherConstructorConstantVariables() (Desktop/Huh/HuhToken
    ↪ -30-10-21.sol#613-1292) uses literals with too many digits:
        - _DEAD_ADDRESS = 0x000000000000000000000000000000000000dEaD (
            ↪ Desktop/Huh/HuhToken-30-10-21.sol#663)
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #too-many-digits

HuhToken._tTotal (Desktop/Huh/HuhToken-30-10-21.sol#621) should be
    ↪ constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #state-variables-that-could-be-declared-constant

renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (Desktop/Huh/HuhToken-30-10-21.sol
            ↪ #72-74)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (Desktop/Huh/HuhToken
            ↪ -30-10-21.sol#80-83)
excludeFromReflection(address) should be declared external:
        - HuhToken.excludeFromReflection(address) (Desktop/Huh/HuhToken
            ↪ -30-10-21.sol#770-773)
transfer(address,uint256) should be declared external:
        - HuhToken.transfer(address,uint256) (Desktop/Huh/HuhToken
            ↪ -30-10-21.sol#907-910)
approve(address,uint256) should be declared external:
        - HuhToken.approve(address,uint256) (Desktop/Huh/HuhToken
            ↪ -30-10-21.sol#912-915)
transferFrom(address,address,uint256) should be declared external:
        - HuhToken.transferFrom(address,address,uint256) (Desktop/Huh/
            ↪ HuhToken-30-10-21.sol#917-924)
increaseAllowance(address,uint256) should be declared external:
        - HuhToken.increaseAllowance(address,uint256) (Desktop/Huh/
            ↪ HuhToken-30-10-21.sol#926-929)
decreaseAllowance(address,uint256) should be declared external:
        - HuhToken.decreaseAllowance(address,uint256) (Desktop/Huh/
            ↪ HuhToken-30-10-21.sol#931-937)
name() should be declared external:
        - HuhToken.name() (Desktop/Huh/HuhToken-30-10-21.sol#942-944)
```

```
symbol() should be declared external:
        - HuhToken.symbol() (Desktop/Huh/HuhToken-30-10-21.sol#946-948)
decimals() should be declared external:
        - HuhToken.decimals() (Desktop/Huh/HuhToken-30-10-21.sol#950-952)
totalSupply() should be declared external:
        - HuhToken.totalSupply() (Desktop/Huh/HuhToken-30-10-21.sol
            ↪ #954-956)
allowance(address,address) should be declared external:
        - HuhToken.allowance(address,address) (Desktop/Huh/HuhToken
            ↪ -30-10-21.sol#964-966)
isExcludedFromReflection(address) should be declared external:
        - HuhToken.isExcludedFromReflection(address) (Desktop/Huh/
            ↪ HuhToken-30-10-21.sol#968-970)
totalFees() should be declared external:
        - HuhToken.totalFees() (Desktop/Huh/HuhToken-30-10-21.sol
            ↪ #972-974)
getTotalCommunityReflection() should be declared external:
        - HuhToken.getTotalCommunityReflection() (Desktop/Huh/HuhToken
            ↪ -30-10-21.sol#987-989)
getTotalNumberOfCommunityReferral() should be declared external:
        - HuhToken.getTotalNumberOfCommunityReferral() (Desktop/Huh/
            ↪ HuhToken-30-10-21.sol#991-993)
getTotalCommunityReferralReward() should be declared external:
        - HuhToken.getTotalCommunityReferralReward() (Desktop/Huh/
            ↪ HuhToken-30-10-21.sol#995-997)
getReferralList(address) should be declared external:
        - HuhToken.getReferralList(address) (Desktop/Huh/HuhToken
            ↪ -30-10-21.sol#999-1001)
getTotalNumberOfUserReferral(address) should be declared external:
        - HuhToken.getTotalNumberOfUserReferral(address) (Desktop/Huh/
            ↪ HuhToken-30-10-21.sol#1003-1005)
getTotalUserReferralReward(address) should be declared external:
        - HuhToken.getTotalUserReferralReward(address) (Desktop/Huh/
            ↪ HuhToken-30-10-21.sol#1007-1009)
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #public-function-that-could-be-declared-external
Desktop/Huh/HuhToken-30-10-21.sol analyzed (10 contracts with 78
    ↪ detectors), 104 result(s) found

//RewardDistributor.sol


RewardDistributor.setRewardTokenAddress(address) (Desktop/Huh/
    ↪ RewardDistributor.sol#564-576) ignores return value by IERC20(
    ↪ rewardToken).transfer(tokenOwner,balance) (Desktop/Huh/
    ↪ RewardDistributor.sol#570)
RewardDistributor.upgradeDistributor(address) (Desktop/Huh/
    ↪ RewardDistributor.sol#583-589) ignores return value by IERC20(
    ↪ rewardToken).transfer(newDistributor,balance) (Desktop/Huh/
    ↪ RewardDistributor.sol#586)
RewardDistributor.emergencyWithdrawTokens(address) (Desktop/Huh/
    ↪ RewardDistributor.sol#623-630) ignores return value by IERC20(
    ↪ token).transfer(tokenOwner,balance) (Desktop/Huh/
    ↪ RewardDistributor.sol#628)
RewardDistributor._giftReward(address,address,uint256) (Desktop/Huh/
    ↪ RewardDistributor.sol#657-670) ignores return value by IERC20(
    ↪ rewardToken).transfer(giftRecipient,amount) (Desktop/Huh/
    ↪ RewardDistributor.sol#666)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #unchecked-transfer

Reentrancy in RewardDistributor._claimRewardInBNB(address) (Desktop/Huh/
    ↪ RewardDistributor.sol#672-683):
        External calls:
        - _swapAndSendBNB(rewardRecipient,referralShares[rewardRecipient
            ↪ ].amount) (Desktop/Huh/RewardDistributor.sol#678)
                - IERC20(rewardToken).approve(address(pcsV2Router),amount)
                    ↪  (Desktop/Huh/RewardDistributor.sol#733)
```

```
            - pcsV2Router.
                ↪ swapExactTokensForETHSupportingFeeOnTransferTokens(
                ↪ amount,0,path,recipient,block.timestamp.add(30)) (
                ↪ Desktop/Huh/RewardDistributor.sol#735-741)
        State variables written after the call(s):
        - referralShares[rewardRecipient].amount = 0 (Desktop/Huh/
            ↪ RewardDistributor.sol#681)
        - referralShares[rewardRecipient].numberOfTimesClaimed =
            ↪ referralShares[rewardRecipient].numberOfTimesClaimed.add
            ↪ (1) (Desktop/Huh/RewardDistributor.sol#682)
Reentrancy in RewardDistributor._claimRewardInBNBToDesiredWallet(address
    ↪ ,address) (Desktop/Huh/RewardDistributor.sol#698-710):
        External calls:
        - _swapAndSendBNB(desiredWallet,referralShares[rewardRecipient].
            ↪ amount) (Desktop/Huh/RewardDistributor.sol#705)
            - IERC20(rewardToken).approve(address(pcsV2Router),amount)
                ↪  (Desktop/Huh/RewardDistributor.sol#733)
            - pcsV2Router.
                ↪ swapExactTokensForETHSupportingFeeOnTransferTokens(
                ↪ amount,0,path,recipient,block.timestamp.add(30)) (
                ↪ Desktop/Huh/RewardDistributor.sol#735-741)
        State variables written after the call(s):
        - referralShares[rewardRecipient].amount = 0 (Desktop/Huh/
            ↪ RewardDistributor.sol#708)
        - referralShares[rewardRecipient].numberOfTimesClaimed =
            ↪ referralShares[rewardRecipient].numberOfTimesClaimed.add
            ↪ (1) (Desktop/Huh/RewardDistributor.sol#709)
Reentrancy in RewardDistributor._claimRewardInDesiredToken(address,
    ↪ address) (Desktop/Huh/RewardDistributor.sol#685-696):
        External calls:
        - _swapAndSendToken(rewardRecipient,referralShares[
            ↪ rewardRecipient].amount,desiredToken) (Desktop/Huh/
            ↪ RewardDistributor.sol#691)
```

```
            - IERC20(rewardToken).approve(address(pcsV2Router),amount)
              ↪  (Desktop/Huh/RewardDistributor.sol#752)
            - pcsV2Router.
              ↪ swapExactTokensForTokensSupportingFeeOnTransferTokens
              ↪ (amount,0,path,recipient,block.timestamp.add(30)) (
              ↪ Desktop/Huh/RewardDistributor.sol#754-760)
        State variables written after the call(s):
        - referralShares[rewardRecipient].amount = 0 (Desktop/Huh/
          ↪ RewardDistributor.sol#694)
        - referralShares[rewardRecipient].numberOfTimesClaimed =
          ↪ referralShares[rewardRecipient].numberOfTimesClaimed.add
          ↪ (1) (Desktop/Huh/RewardDistributor.sol#695)
    Reentrancy in RewardDistributor.
    ↪ _claimRewardInDesiredTokenToDesiredWallet(address,address,address
    ↪ ) (Desktop/Huh/RewardDistributor.sol#712-724):
        External calls:
        - _swapAndSendToken(desiredWallet,referralShares[rewardRecipient
          ↪ ].amount,desiredToken) (Desktop/Huh/RewardDistributor.sol
          ↪ #719)
            - IERC20(rewardToken).approve(address(pcsV2Router),amount)
              ↪  (Desktop/Huh/RewardDistributor.sol#752)
            - pcsV2Router.
              ↪ swapExactTokensForTokensSupportingFeeOnTransferTokens
              ↪ (amount,0,path,recipient,block.timestamp.add(30)) (
              ↪ Desktop/Huh/RewardDistributor.sol#754-760)
        State variables written after the call(s):
        - referralShares[rewardRecipient].amount = 0 (Desktop/Huh/
          ↪ RewardDistributor.sol#722)
        - referralShares[rewardRecipient].numberOfTimesClaimed =
          ↪ referralShares[rewardRecipient].numberOfTimesClaimed.add
          ↪ (1) (Desktop/Huh/RewardDistributor.sol#723)
    Reentrancy in RewardDistributor._giftReward(address,address,uint256) (
    ↪ Desktop/Huh/RewardDistributor.sol#657-670):
        External calls:
```

```
        - IERC20(rewardToken).approve(rewardRecipient,amount) (Desktop/
            ↪ Huh/RewardDistributor.sol#665)
        - IERC20(rewardToken).transfer(giftRecipient,amount) (Desktop/Huh
            ↪ /RewardDistributor.sol#666)
        State variables written after the call(s):
        - referralShares[rewardRecipient].amount = referralShares[
            ↪ rewardRecipient].amount.sub(amount) (Desktop/Huh/
            ↪ RewardDistributor.sol#669)
Reentrancy in RewardDistributor.setRewardTokenAddress(address) (Desktop/
    ↪ Huh/RewardDistributor.sol#564-576):
        External calls:
        - IERC20(rewardToken).transfer(tokenOwner,balance) (Desktop/Huh/
            ↪ RewardDistributor.sol#570)
        State variables written after the call(s):
        - rewardToken = _rewardToken (Desktop/Huh/RewardDistributor.sol
            ↪ #574)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #reentrancy-vulnerabilities-1

RewardDistributor._giftReward(address,address,uint256) (Desktop/Huh/
    ↪ RewardDistributor.sol#657-670) ignores return value by IERC20(
    ↪ rewardToken).approve(rewardRecipient,amount) (Desktop/Huh/
    ↪ RewardDistributor.sol#665)
RewardDistributor._swapAndSendBNB(address,uint256) (Desktop/Huh/
    ↪ RewardDistributor.sol#726-742) ignores return value by IERC20(
    ↪ rewardToken).approve(address(pcsV2Router),amount) (Desktop/Huh/
    ↪ RewardDistributor.sol#733)
RewardDistributor._swapAndSendToken(address,uint256,address) (Desktop/
    ↪ Huh/RewardDistributor.sol#744-761) ignores return value by IERC20
    ↪ (rewardToken).approve(address(pcsV2Router),amount) (Desktop/Huh/
    ↪ RewardDistributor.sol#752)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #unused-return
```

```
RewardDistributor.transferTokenOwnership(address).newOwner (Desktop/Huh/
    ↪ RewardDistributor.sol#578) lacks a zero-check on :
                - tokenOwner = newOwner (Desktop/Huh/RewardDistributor.sol
                    ↪ #579)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #missing-zero-address-validation

Reentrancy in RewardDistributor.claimReward() (Desktop/Huh/
    ↪ RewardDistributor.sol#637-640):
        External calls:
        - _claimRewardInBNB(msg.sender) (Desktop/Huh/RewardDistributor.
            ↪ sol#638)
                - IERC20(rewardToken).approve(address(pcsV2Router),amount)
                    ↪  (Desktop/Huh/RewardDistributor.sol#733)
                - pcsV2Router.
                    ↪ swapExactTokensForETHSupportingFeeOnTransferTokens(
                    ↪ amount,0,path,recipient,block.timestamp.add(30)) (
                    ↪ Desktop/Huh/RewardDistributor.sol#735-741)
        Event emitted after the call(s):
        - ClaimReward(msg.sender) (Desktop/Huh/RewardDistributor.sol#639)
Reentrancy in RewardDistributor.claimRewardInDesiredToken(address) (
    ↪ Desktop/Huh/RewardDistributor.sol#642-645):
        External calls:
        - _claimRewardInDesiredToken(msg.sender,desiredToken) (Desktop/
            ↪ Huh/RewardDistributor.sol#643)
                - IERC20(rewardToken).approve(address(pcsV2Router),amount)
                    ↪  (Desktop/Huh/RewardDistributor.sol#752)
                - pcsV2Router.
                    ↪ swapExactTokensForTokensSupportingFeeOnTransferTokens
                    ↪ (amount,0,path,recipient,block.timestamp.add(30)) (
                    ↪ Desktop/Huh/RewardDistributor.sol#754-760)
        Event emitted after the call(s):
        - ClaimRewardInDesiredToken(msg.sender,desiredToken) (Desktop/Huh
            ↪ /RewardDistributor.sol#644)
```

```
Reentrancy in RewardDistributor.claimRewardInDesiredTokenToDesiredWallet
    ↪ (address,address) (Desktop/Huh/RewardDistributor.sol#652-655):
        External calls:
        - _claimRewardInDesiredTokenToDesiredWallet(msg.sender,
            ↪ desiredWallet,desiredToken) (Desktop/Huh/RewardDistributor
            ↪ .sol#653)
                - IERC20(rewardToken).approve(address(pcsV2Router),amount)
                    ↪  (Desktop/Huh/RewardDistributor.sol#752)
                - pcsV2Router.
                    ↪ swapExactTokensForTokensSupportingFeeOnTransferTokens
                    ↪ (amount,0,path,recipient,block.timestamp.add(30)) (
                    ↪ Desktop/Huh/RewardDistributor.sol#754-760)
        Event emitted after the call(s):
        - ClaimRewardInDesiredTokenToDesiredWallet(msg.sender,
            ↪ desiredWallet,desiredToken) (Desktop/Huh/RewardDistributor
            ↪ .sol#654)
Reentrancy in RewardDistributor.claimRewardToDesiredWallet(address) (
    ↪ Desktop/Huh/RewardDistributor.sol#647-650):
        External calls:
        - _claimRewardInBNBToDesiredWallet(msg.sender,desiredWallet) (
            ↪ Desktop/Huh/RewardDistributor.sol#648)
                - IERC20(rewardToken).approve(address(pcsV2Router),amount)
                    ↪  (Desktop/Huh/RewardDistributor.sol#733)
                - pcsV2Router.
                    ↪ swapExactTokensForETHSupportingFeeOnTransferTokens(
                    ↪ amount,0,path,recipient,block.timestamp.add(30)) (
                    ↪ Desktop/Huh/RewardDistributor.sol#735-741)
        Event emitted after the call(s):
        - ClaimRewardToDesiredWallet(msg.sender,desiredWallet) (Desktop/
            ↪ Huh/RewardDistributor.sol#649)
Reentrancy in RewardDistributor.giftReward(address,uint256) (Desktop/Huh
    ↪ /RewardDistributor.sol#632-635):
        External calls:
```

```
        - _giftReward(msg.sender,giftRecipient,amount) (Desktop/Huh/
            ↪ RewardDistributor.sol#633)
                - IERC20(rewardToken).approve(rewardRecipient,amount) (
                    ↪ Desktop/Huh/RewardDistributor.sol#665)
                - IERC20(rewardToken).transfer(giftRecipient,amount) (
                    ↪ Desktop/Huh/RewardDistributor.sol#666)
        Event emitted after the call(s):
        - GiftReward(msg.sender,giftRecipient,amount) (Desktop/Huh/
            ↪ RewardDistributor.sol#634)
Reentrancy in RewardDistributor.setRewardTokenAddress(address) (Desktop/
    ↪ Huh/RewardDistributor.sol#564-576):
        External calls:
        - IERC20(rewardToken).transfer(tokenOwner,balance) (Desktop/Huh/
            ↪ RewardDistributor.sol#570)
        Event emitted after the call(s):
        - SetRewardTokenAddress(_rewardToken) (Desktop/Huh/
            ↪ RewardDistributor.sol#575)
Reentrancy in RewardDistributor.upgradeDistributor(address) (Desktop/Huh
    ↪ /RewardDistributor.sol#583-589):
        External calls:
        - IERC20(rewardToken).transfer(newDistributor,balance) (Desktop/
            ↪ Huh/RewardDistributor.sol#586)
        Event emitted after the call(s):
        - UpgradeDistributor(newDistributor) (Desktop/Huh/
            ↪ RewardDistributor.sol#587)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #reentrancy-vulnerabilities-3

Address.isContract(address) (Desktop/Huh/RewardDistributor.sol#170-179)
    ↪ uses assembly
        - INLINE ASM (Desktop/Huh/RewardDistributor.sol#177)
Address._functionCallWithValue(address,bytes,uint256,string) (Desktop/
    ↪ Huh/RewardDistributor.sol#263-284) uses assembly
        - INLINE ASM (Desktop/Huh/RewardDistributor.sol#276-279)
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #assembly-usage

Address._functionCallWithValue(address,bytes,uint256,string) (Desktop/
    ↪ Huh/RewardDistributor.sol#263-284) is never used and should be
    ↪ removed
Address.functionCall(address,bytes) (Desktop/Huh/RewardDistributor.sol
    ↪ #223-225) is never used and should be removed
Address.functionCall(address,bytes,string) (Desktop/Huh/
    ↪ RewardDistributor.sol#233-235) is never used and should be
    ↪ removed
Address.functionCallWithValue(address,bytes,uint256) (Desktop/Huh/
    ↪ RewardDistributor.sol#248-250) is never used and should be
    ↪ removed
Address.functionCallWithValue(address,bytes,uint256,string) (Desktop/Huh
    ↪ /RewardDistributor.sol#258-261) is never used and should be
    ↪ removed
Address.isContract(address) (Desktop/Huh/RewardDistributor.sol#170-179)
    ↪ is never used and should be removed
Address.sendValue(address,uint256) (Desktop/Huh/RewardDistributor.sol
    ↪ #197-203) is never used and should be removed
SafeMath.div(uint256,uint256) (Desktop/Huh/RewardDistributor.sol#93-95)
    ↪ is never used and should be removed
SafeMath.div(uint256,uint256,string) (Desktop/Huh/RewardDistributor.sol
    ↪ #109-115) is never used and should be removed
SafeMath.mod(uint256,uint256) (Desktop/Huh/RewardDistributor.sol
    ↪ #129-131) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (Desktop/Huh/RewardDistributor.sol
    ↪ #145-148) is never used and should be removed
SafeMath.mul(uint256,uint256) (Desktop/Huh/RewardDistributor.sol#67-79)
    ↪ is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #dead-code
```

```
Low level call in Address.sendValue(address,uint256) (Desktop/Huh/
    ↪ RewardDistributor.sol#197-203):
        - (success) = recipient.call{value: amount}() (Desktop/Huh/
            ↪ RewardDistributor.sol#201)
Low level call in Address._functionCallWithValue(address,bytes,uint256,
    ↪ string) (Desktop/Huh/RewardDistributor.sol#263-284):
        - (success,returndata) = target.call{value: weiValue}(data) (
            ↪ Desktop/Huh/RewardDistributor.sol#267)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #low-level-calls

Function IUniswapV2Router01.WETH() (Desktop/Huh/RewardDistributor.sol
    ↪ #290) is not in mixedCase
Parameter RewardDistributor.setRewardTokenAddress(address)._rewardToken
    ↪ (Desktop/Huh/RewardDistributor.sol#564) is not in mixedCase
Constant RewardDistributor.router (Desktop/Huh/RewardDistributor.sol
    ↪ #533) is not in UPPER_CASE_WITH_UNDERSCORES
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #conformance-to-solidity-naming-conventions

Variable IUniswapV2Router01.addLiquidity(address,address,uint256,uint256
    ↪ ,uint256,uint256,address,uint256).amountADesired (Desktop/Huh/
    ↪ RewardDistributor.sol#295) is too similar to IUniswapV2Router01.
    ↪ addLiquidity(address,address,uint256,uint256,uint256,uint256,
    ↪ address,uint256).amountBDesired (Desktop/Huh/RewardDistributor.
    ↪ sol#296)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
    ↪ #variable-names-are-too-similar
Desktop/Huh/RewardDistributor.sol analyzed (8 contracts with 78
    ↪ detectors), 41 result(s) found
```

Conclusion:

Most of the vulnerabilities found by the analysis have already been addressed by the smart contract code review.

5 Conclusion

In this audit, we examined the design and implementation of HuhToken contract and discovered several issues of varying severity. HuhToken team addressed 7 issues raised in the initial report and implemented the necessary fixes, while classifying the rest as a risk with low-probability of occurrence. Shellboxes' auditors advised HuhToken Team to maintain a high level of vigilance and to keep those findings in mind in order to avoid any future complications.

For a Contract Audit, contact us at contact@shellboxes.com