HALBORN

LimeChain - HeliSwap

Smart Contract Security Audit

Prepared by: **Halborn**

Date of Engagement: **July 11th, 2022 - July 29th, 2022**

Visit: **Halborn.com**

DOCUMENT REVISION HISTORY

VERSION	MODIFICATION	DATE	AUTHOR
0.1	Document Creation	07/29/2022	Pawel Bartunek
0.2	Draft Review	08/01/2022	Kubilay Onur Gungor
0.3	Draft Review	08/01/2022	Gabi Urrutia
1.0	Remediation Plan	08/09/2022	Pawel Bartunek
1.1	Draft Review	08/09/2022	Kubilay Onur Gungor
1.2	Draft Review	08/09/2022	Gabi Urrutia

CONTACTS

CONTACT	COMPANY	EMAIL
Rob Behnke	Halborn	Rob.Behnke@halborn.com
Steven Walbroehl	Halborn	Steven.Walbroehl@halborn.com
Gabi Urrutia	Halborn	Gabi.Urrutia@halborn.com
Kubilay Onur Gungor	Halborn	Kubilay.Gungor@halborn.com
Pawel Bartunek	Halborn	Pawel.Bartunek@halborn.com

EXECUTIVE OVERVIEW

1.1 INTRODUCTION

LimeChain engaged Halborn to conduct a security audit on their smart contracts beginning on July 11th, 2022 and ending on July 29th, 2022 . The security assessment was scoped to the smart contracts provided to the Halborn team.

1.2 AUDIT SUMMARY

The team at Halborn was provided three weeks for the engagement and assigned a full-time security engineer to audit the security of the smart contract. The security engineer is a blockchain and smart-contract security expert with advanced penetration testing, smart-contract hacking, and deep knowledge of multiple blockchain protocols.

The purpose of this audit is to:

- Ensure that smart contract functions operate as intended
- Identify potential security issues with the smart contracts

In summary, Halborn identified some improvements to reduce the likelihood and impact of risks, which have been mostly addressed by the LimeChain .

1.3 TEST APPROACH & METHODOLOGY

Halborn performed a combination of manual and automated security testing to balance efficiency, timeliness, practicality, and accuracy in regard to the scope of this audit. While manual testing is recommended to uncover flaws in logic, process, and implementation; automated testing techniques help enhance coverage of the bridge code and can quickly identify items that do not follow security best practices. The following phases and associated tools were used throughout the term of the audit:

- Research into architecture and purpose
- Smart contract manual code review and walk-through
- Graphing out functionality and contract logic/connectivity/functions (solgraph)
- Manual assessment of use and safety for the critical Solidity variables and functions in scope to identify any arithmetic related vulnerability classes
- Manual testing by custom scripts
- Static Analysis of security for scoped contract, and imported functions. (Slither)
- Local and Testnet deployment (Hardhat, Remix IDE)

RISK METHODOLOGY:

Vulnerabilities or issues observed by Halborn are ranked based on the risk assessment methodology by measuring the **LIKELIHOOD** of a security incident and the **IMPACT** should an incident occur. This framework works for communicating the characteristics and impacts of technology vulnerabilities. The quantitative model ensures repeatable and accurate measurement while enabling users to see the underlying vulnerability characteristics that were used to generate the Risk scores. For every vulnerability, a risk level will be calculated on a scale of 5 to 1 with 5 being the highest likelihood or impact.

RISK SCALE - LIKELIHOOD

5 - Almost certain an incident will occur.
4 - High probability of an incident occurring.
3 - Potential of a security incident in the long term.
2 - Low probability of an incident occurring.
1 - Very unlikely issue will cause an incident.

RISK SCALE - IMPACT

5 - May cause devastating and unrecoverable impact or loss.
4 - May cause a significant level of impact or loss.
3 - May cause a partial impact or loss to many.
2 - May cause temporary impact or loss.

1 - May cause minimal or un-noticeable impact.

The risk level is then calculated using a sum of these two values, creating a value of 10 to 1 with 10 being the highest level of security risk.

10 - CRITICAL
9 - 8 - HIGH
7 - 6 - MEDIUM
5 - 4 - LOW
3 - 1 - VERY LOW AND INFORMATIONAL

1.4 SCOPE

IN-SCOPE:
The security assessment was scoped to the following smart contracts:

core:
- UniswapV2ERC20.sol
- UniswapV2Factory.sol
- UniswapV2Pair.sol

core/interfaces:
- IERC20.sol
- IUniswapV2Callee.sol
- IUniswapV2ERC20.sol
- IUniswapV2Factory.sol
- IUniswapV2Pair.sol

core/libraries:
- Math.sol
- SafeMath.sol
- UQ112x112.sol

mock:
- ERC20.sol
- MockToken.sol
- MockWHBAR.sol
- RouterEventEmitter.sol
- TestUniswapV2ERC20.sol

periphery:
- UniswapV2Router02.sol

periphery/interfaces:
- IERC20.sol
- IUniswapV2Router01.sol
- IUniswapV2Router02.sol
- IWHBAR.sol

periphery/libraries:
- SafeMath.sol
- TransferHelper.sol
- UniswapV2Library.sol

Commit ID: 1649c6c3909362639be62b3eb10671c5817f9cf9

Remediation Commit ID: 442bf2e8136dcf93cf4343675b64d05163544d44

2. ASSESSMENT SUMMARY & FINDINGS OVERVIEW

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL
0	0	0	2	2

LIKELIHOOD

IMPACT

(HAL-02)				
	(HAL-01)			
(HAL-03) (HAL-04)				

EXECUTIVE OVERVIEW

SECURITY ANALYSIS	RISK LEVEL	REMEDIATION DATE
HAL01 - MISSING ADDRESS CHECK	Low	SOLVED - 08/09/2022
HAL02 - PRAGMA VERSION DEPRECATED	Low	RISK ACCEPTED
HAL03 - OPEN TODOs	Informational	SOLVED - 08/09/2022
HAL04 - FUNCTION NAMES WERE NOT UPDATED WITH WHBAR	Informational	SOLVED - 08/09/2022

FINDINGS & TECH DETAILS

3.1 (HAL-01) MISSING ADDRESS CHECK - LOW

Description:

The constructor and setFeeToSetter methods are not validating the _feeToSetter parameter. This address can be set to 0x0, which makes future fee setting changes impossible (because of the require statement on line 84).

Code Location:

Listing 1: core/UniswapV2Factory.sol (Line 34)

```
33 constructor(address _feeToSetter) public {
34     feeToSetter = _feeToSetter;
35 }
```

Listing 2: core/UniswapV2Factory.sol (Line 85)

```
83 function setFeeToSetter(address _feeToSetter) external {
84     require(msg.sender == feeToSetter, 'UniswapV2: FORBIDDEN');
85     feeToSetter = _feeToSetter;
86 }
```

Risk Level:

Likelihood - 2
Impact - 2

Recommendation:

Adding proper address validation when each state variable assignment is done from user-provided input will help increase the security posture.

Remediation Plan:

SOLVED: The \client team added an address validation to the constructor.

Being able to set feeToSetter to address 0x0 via the setter function is a design decision. The client stated that:

“We think that having the option to forfeit this role to the zero address is something that may be utilized at some point. That is why the setFeeToSetter has the option to change the feeToSetter to the 0x0 address.”

3.2 (HAL-02) PRAGMA VERSION DEPRECATED - LOW

Description:

Contracts use older versions of Solidity 0.5.16 and 0.6.6.
While these versions are still functional, and most security issues are safely implemented by mitigating contracts with utility libraries such as SafeMath.sol, it increases the risk to long-term sustainability and code integrity of Solidity.

Code Location:

Listing 3

```
core/UniswapV2Factory.sol:pragma solidity =0.5.16;
core/UniswapV2ERC20.sol:pragma solidity =0.5.16;
core/libraries/SafeMath.sol:pragma solidity =0.5.16;
core/libraries/UQ112x112.sol:pragma solidity =0.5.16;
core/libraries/Math.sol:pragma solidity =0.5.16;
core/UniswapV2Pair.sol:pragma solidity =0.5.16;
mock/RouterEventEmitter.sol:pragma solidity =0.6.6;
mock/ERC20.sol:pragma solidity =0.6.6;
mock/TestUniswapV2ERC20.sol:pragma solidity =0.5.16;
periphery/libraries/SafeMath.sol:pragma solidity =0.6.6;
periphery/UniswapV2Router02.sol:pragma solidity =0.6.6;
```

Risk Level:

Likelihood - 1
Impact - 3

Recommendation:

At the time of this audit, the current version is already 0.8.15. When possible, it is strongly recommended to use the most up-to-date and tested

pragma versions to take advantage of new features that provide checks and accounting, as well as to avoid insecure code usage.

Remediation Plan:

RISK ACCEPTED: The \client team decided not to resolve this issue, as it could lead to more uncertainties than benefits.

3.3 (HAL-03) OPEN TODOs - INFORMATIONAL

Description:

Open TODOs can point to architectural or programming issues that still need to be resolved. Often these kinds of comments indicate areas of complexity or confusion for developers. This provides value and insight to an attacker who is aiming to cause damage to the protocol.

Code Location:

Listing 4

```
% grep -R TODO -n *
core/UniswapV2Pair.sol:56:        // TODO: check if data is eq 22
core/interfaces/IUniswapV2Pair.sol:32:    // TODO revert back the
↳ Mint/Burn changes
```

Risk Level:

Likelihood - 1
Impact - 1

Recommendation:

Consider resolving TODOs before deploying code to production. It is recommended to use a separate issue tracker or other project management software to track development tasks.

Remediation Plan:

SOLVED: The issue was solved, the TODOs were removed from the code base:

- IUniswapV2Pair

- UniswapV2Pair

Listing 5

```
1 $ grep -R TODO -n *
```

3.4 (HAL-04) FUNCTION NAMES WERE NOT UPDATED WITH WHBAR - INFORMATIONAL

Description:

The names of the functions, which are used to trade in ETH, still contain ETH in their names. Such a naming convention can be misleading for HeliSwap users, since the protocol supports HBAR instead of ETH.

Code Location:

Listing 6

```
mock/RouterEventEmitter.sol:40:    function swapExactETHForTokens(
mock/RouterEventEmitter.sol:54:    function swapTokensForExactETH(
mock/RouterEventEmitter.sol:69:    function swapExactTokensForETH(
mock/RouterEventEmitter.sol:84:    function swapETHForExactTokens(
periphery/libraries/TransferHelper.sol:47:    function
↳ safeTransferETH(address to, uint256 value) internal {
periphery/interfaces/IUniswapV2Router01.sol:17:    function
↳ addLiquidityETH(
periphery/interfaces/IUniswapV2Router01.sol:34:    function
↳ removeLiquidityETH(
periphery/interfaces/IUniswapV2Router01.sol:56:    function
↳ swapExactETHForTokens(uint amountOutMin, address[] calldata path,
↳ address to, uint deadline)
periphery/interfaces/IUniswapV2Router01.sol:60:    function
↳ swapTokensForExactETH(uint amountOut, uint amountInMax, address[]
↳ calldata path, address to, uint deadline)
periphery/interfaces/IUniswapV2Router01.sol:63:    function
↳ swapExactTokensForETH(uint amountIn, uint amountOutMin, address[]
↳ calldata path, address to, uint deadline)
periphery/interfaces/IUniswapV2Router01.sol:66:    function
↳ swapETHForExactTokens(uint amountOut, address[] calldata path,
↳ address to, uint deadline)
periphery/interfaces/IUniswapV2Router02.sol:6:    function
↳ removeLiquidityETHSupportingFeeOnTransferTokens(
periphery/interfaces/IUniswapV2Router02.sol:21:    function
↳ swapExactETHForTokensSupportingFeeOnTransferTokens(
periphery/interfaces/IUniswapV2Router02.sol:27:    function
↳ swapExactTokensForETHSupportingFeeOnTransferTokens(
```

```
15 periphery/UniswapV2Router02.sol:90:    function addLiquidityETH(
16 periphery/UniswapV2Router02.sol:134:    function
↳ removeLiquidityETH(
17 periphery/UniswapV2Router02.sol:163:    function
↳ removeLiquidityETHSupportingFeeOnTransferTokens(
18 periphery/UniswapV2Router02.sol:234:    function
↳ swapExactETHForTokens(uint amountOutMin, address[] calldata path,
↳ address to, uint deadline)
19 periphery/UniswapV2Router02.sol:249:    function
↳ swapTokensForExactETH(uint amountOut, uint amountInMax, address[]
↳ calldata path, address to, uint deadline)
20 periphery/UniswapV2Router02.sol:266:    function
↳ swapExactTokensForETH(uint amountIn, uint amountOutMin, address[]
↳ calldata path, address to, uint deadline)
21 periphery/UniswapV2Router02.sol:283:    function
↳ swapETHForExactTokens(uint amountOut, address[] calldata path,
↳ address to, uint deadline)
22 periphery/UniswapV2Router02.sol:338:    function
↳ swapExactETHForTokensSupportingFeeOnTransferTokens(
23 periphery/UniswapV2Router02.sol:361:    function
↳ swapExactTokensForETHSupportingFeeOnTransferTokens(
```

Risk Level:

Likelihood - 1
Impact - 1

Recommendation:

Since contracts support HBAR/WHBAR, it is recommended to update function names with HBAR instead of ETH.

Remediation Plan:

SOLVED: All function names were updated from ETH to HBAR.

MANUAL TESTING

MANUAL TESTING

Halborn performed several manual tests on the HeliSwap contracts using the provided utility scripts and modifying the provided Hardhat test suite. Tests were run on Hedera local node.

HeliSwap contracts are a modified version of Uniswap V2 contracts, with some adjustments required by Hedera Hashgraph. Manual tests were focused on the swap functionality:

```
 % npx hardhat --network local test test/UniswapV2Pair.audit.spec.ts
Deprecated: Use `ledgerId` instead


  UniswapV2Pair HTS - HTS
Deprecated: Use `ledgerId` instead
    ✔ getInputPrice:0 (3087ms)
    ✔ getInputPrice:1 (3087ms)
    ✔ getInputPrice:2 (3082ms)
    ✔ getInputPrice:3 (3083ms)
    ✔ getInputPrice:4 (3093ms)
    ✔ getInputPrice:5 (3088ms)
    ✔ getInputPrice:6 (3080ms)
    ✔ optimistic:0 (3095ms)
    ✔ optimistic:1 (3084ms)
    ✔ optimistic:2 (3090ms)
    ✔ swap:token0 (5781ms)
    ✔ swap:token1 (5654ms)

  UniswapV2Pair Non-HTS - HTS
    ✔ getInputPrice:0 (3077ms)
    ✔ getInputPrice:1 (3076ms)
    ✔ getInputPrice:2 (3092ms)
    ✔ getInputPrice:3 (3089ms)
    ✔ getInputPrice:4 (3079ms)
    ✔ getInputPrice:5 (3090ms)
    ✔ getInputPrice:6 (3098ms)
    ✔ optimistic:0 (3071ms)
    ✔ optimistic:1 (3075ms)
    ✔ optimistic:2 (3081ms)
    ✔ swap:token0 (5651ms)
    ✔ swap:token1 (5640ms)

  UniswapV2Pair HTS - Non-HTS
    ✔ getInputPrice:0 (3086ms)
    ✔ getInputPrice:1 (3081ms)
    ✔ getInputPrice:2 (3077ms)
    ✔ getInputPrice:3 (3092ms)
    ✔ getInputPrice:4 (3092ms)
    ✔ getInputPrice:5 (3076ms)
    ✔ getInputPrice:6 (3091ms)
    ✔ optimistic:0 (3071ms)
    ✔ optimistic:1 (3090ms)
    ✔ optimistic:2 (3079ms)
    ✔ swap:token0 (5648ms)
    ✔ swap:token1 (5699ms)

  UniswapV2Pair Non-HTS - Non-HTS
    ✔ getInputPrice:0 (3075ms)
    ✔ getInputPrice:1 (3085ms)
    ✔ getInputPrice:2 (3079ms)
    ✔ getInputPrice:3 (3081ms)
    ✔ getInputPrice:4 (3081ms)
    ✔ getInputPrice:5 (3081ms)
    ✔ getInputPrice:6 (3079ms)
    ✔ optimistic:0 (3074ms)
    ✔ optimistic:1 (3083ms)
    ✔ optimistic:2 (3099ms)
    ✔ swap:token0 (5661ms)
    ✔ swap:token1 (5639ms)


  48 passing (8m)
```

No issues were found during manual test.

AUTOMATED TESTING

5.1 STATIC ANALYSIS REPORT

Description:

Halborn used automated testing techniques to enhance the coverage of certain areas of the scoped contracts. Among the tools used was Slither, a Solidity static analysis framework. After Halborn verified all the contracts in the repository and was able to compile them correctly into their ABI and binary formats, Slither was run on the all-scoped contracts. This tool can statically verify mathematical relationships between Solidity variables to detect invalid or inconsistent usage of the contracts' APIs across the entire code-base.

Slither results:

contracts/core/UniswapV2ERC20.sol:

Listing 7

```
UniswapV2ERC20.permit(address,address,uint256,uint256,uint8,
↳ bytes32,bytes32) (contracts/core/UniswapV2ERC20.sol#81-93) uses
↳ timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(deadline >= block.timestamp,
↳ UniswapV2: EXPIRED) (contracts/core/UniswapV2ERC20.sol#82)
Reference: https://github.com/crytic/slither/wiki/Detector-
↳ Documentation#block-timestamp

UniswapV2ERC20.constructor() (contracts/core/UniswapV2ERC20.sol
↳ #24-38) uses assembly
        - INLINE ASM (contracts/core/UniswapV2ERC20.sol#26-28)
Reference: https://github.com/crytic/slither/wiki/Detector-
↳ Documentation#assembly-usage

Different versions of Solidity are used:
        - Version used: ['=0.5.16', '>=0.5.0']
        - =0.5.16 (contracts/core/UniswapV2ERC20.sol#1)
        - >=0.5.0 (contracts/core/interfaces/IUniswapV2ERC20.sol
↳ #1)
        - =0.5.16 (contracts/core/libraries/SafeMath.sol#1)
```

```
15 Reference: https://github.com/crytic/slither/wiki/Detector-
↳ Documentation#different-pragma-directives-are-used
16
17 SafeMath.mul(uint256,uint256) (contracts/core/libraries/SafeMath.
↳ sol#14-16) is never used and should be removed
18 UniswapV2ERC20._burn(address,uint256) (contracts/core/
↳ UniswapV2ERC20.sol#46-50) is never used and should be removed
19 UniswapV2ERC20._mint(address,uint256) (contracts/core/
↳ UniswapV2ERC20.sol#40-44) is never used and should be removed
20 Reference: https://github.com/crytic/slither/wiki/Detector-
↳ Documentation#dead-code
21
22 Pragma version>=0.5.0 (contracts/core/interfaces/IUniswapV2ERC20.
↳ sol#1) allows old versions
23 Reference: https://github.com/crytic/slither/wiki/Detector-
↳ Documentation#incorrect-versions-of-solidity
24
25 Variable UniswapV2ERC20.DOMAIN_SEPARATOR (contracts/core/
↳ UniswapV2ERC20.sol#16) is not in mixedCase
26 Function IUniswapV2ERC20.DOMAIN_SEPARATOR() (contracts/core/
↳ interfaces/IUniswapV2ERC20.sol#18) is not in mixedCase
27 Function IUniswapV2ERC20.PERMIT_TYPEHASH() (contracts/core/
↳ interfaces/IUniswapV2ERC20.sol#19) is not in mixedCase
28 Reference: https://github.com/crytic/slither/wiki/Detector-
↳ Documentation#conformance-to-solidity-naming-conventions
29 contracts/core/UniswapV2ERC20.sol analyzed (3 contracts with 78
↳ detectors), 10 result(s) found
```

contracts/core/UniswapV2Factory.sol:

Listing 8

```
1 Reentrancy in UniswapV2Factory.createPair(address,address) (
↳ contracts/core/UniswapV2Factory.sol#41-76):
2         External calls:
3         - IUniswapV2Pair(pair).initialize(token0,token1) (
↳ contracts/core/UniswapV2Factory.sol#51)
4         State variables written after the call(s):
5         - getPair[token0][token1] = pair (contracts/core/
↳ UniswapV2Factory.sol#52)
6         - getPair[token1][token0] = pair (contracts/core/
↳ UniswapV2Factory.sol#53)
7
```

```
UniswapV2Factory.constructor(address)._feeToSetter (contracts/core/UniswapV2Factory.sol#33) lacks a zero-check on :
                - feeToSetter = _feeToSetter (contracts/core/UniswapV2Factory.sol#34)
UniswapV2Factory.setFeeTo(address)._feeTo (contracts/core/UniswapV2Factory.sol#78) lacks a zero-check on :
                - feeTo = _feeTo (contracts/core/UniswapV2Factory.sol#80)
UniswapV2Factory.setFeeToSetter(address)._feeToSetter (contracts/core/UniswapV2Factory.sol#83) lacks a zero-check on :
                - feeToSetter = _feeToSetter (contracts/core/UniswapV2Factory.sol#85)

Reentrancy in UniswapV2Factory.createPair(address,address) (contracts/core/UniswapV2Factory.sol#41-76):
        External calls:
        - IUniswapV2Pair(pair).initialize(token0,token1) (contracts/core/UniswapV2Factory.sol#51)
        State variables written after the call(s):
        - allPairs.push(pair) (contracts/core/UniswapV2Factory.sol#54)
Reentrancy in UniswapV2Pair.swap(uint256,uint256,address,bytes) (contracts/core/UniswapV2Pair.sol#182-210):
        External calls:
        - _safeTransfer(_token0,to,amount0Out) (contracts/core/UniswapV2Pair.sol#193)
                - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (contracts/core/UniswapV2Pair.sol#54)
        - _safeTransfer(_token1,to,amount1Out) (contracts/core/UniswapV2Pair.sol#194)
                - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (contracts/core/UniswapV2Pair.sol#54)
        - IUniswapV2Callee(to).uniswapV2Call(msg.sender,amount0Out,amount1Out,data) (contracts/core/UniswapV2Pair.sol#195)
        State variables written after the call(s):
        - _update(balance0,balance1,_reserve0,_reserve1) (contracts/core/UniswapV2Pair.sol#208)
                - price0CumulativeLast += uint256(UQ112x112.encode(_reserve1).uqdiv(_reserve0)) * timeElapsed (contracts/core/UniswapV2Pair.sol#102)
        - _update(balance0,balance1,_reserve0,_reserve1) (contracts/core/UniswapV2Pair.sol#208)
```

```
31                 - price1CumulativeLast += uint256(UQ112x112.encode
↳ (_reserve0).uqdiv(_reserve1)) * timeElapsed (contracts/core/
↳ UniswapV2Pair.sol#103)
32 Reference: https://github.com/crytic/slither/wiki/Detector-
↳ Documentation#reentrancy-vulnerabilities-2
33
34 Reentrancy in UniswapV2Factory.createPair(address,address) (
↳ contracts/core/UniswapV2Factory.sol#41-76):
35         External calls:
36         - IUniswapV2Pair(pair).initialize(token0,token1) (
↳ contracts/core/UniswapV2Factory.sol#51)
37         Event emitted after the call(s):
38         - PairCreated(token0,token1,pair,allPairs.length,
↳ token0Symbol,token1Symbol,token0Name,token1Name,token0Decimals,
↳ token1Decimals) (contracts/core/UniswapV2Factory.sol#64-75)
39 Reentrancy in UniswapV2Pair.swap(uint256,uint256,address,bytes) (
↳ contracts/core/UniswapV2Pair.sol#182-210):
40         External calls:
41         - _safeTransfer(_token0,to,amount0Out) (contracts/core/
↳ UniswapV2Pair.sol#193)
42                 - (success,data) = token.call(abi.
↳ encodeWithSelector(SELECTOR,to,value)) (contracts/core/
↳ UniswapV2Pair.sol#54)
43         - _safeTransfer(_token1,to,amount1Out) (contracts/core/
↳ UniswapV2Pair.sol#194)
44                 - (success,data) = token.call(abi.
↳ encodeWithSelector(SELECTOR,to,value)) (contracts/core/
↳ UniswapV2Pair.sol#54)
45         - IUniswapV2Callee(to).uniswapV2Call(msg.sender,amount0Out
↳ ,amount1Out,data) (contracts/core/UniswapV2Pair.sol#195)
46         Event emitted after the call(s):
47         - Swap(msg.sender,amount0In,amount1In,amount0Out,
↳ amount1Out,to) (contracts/core/UniswapV2Pair.sol#209)
48         - Sync(reserve0,reserve1,totalSupply) (contracts/core/
↳ UniswapV2Pair.sol#108)
49                 - _update(balance0,balance1,_reserve0,_reserve1) (
↳ contracts/core/UniswapV2Pair.sol#208)
50 Reference: https://github.com/crytic/slither/wiki/Detector-
↳ Documentation#reentrancy-vulnerabilities-3
51 UniswapV2Factory.createPair(address,address) (contracts/core/
↳ UniswapV2Factory.sol#41-76) uses assembly
52         - INLINE ASM (contracts/core/UniswapV2Factory.sol#48-50)
53 Reference: https://github.com/crytic/slither/wiki/Detector-
↳ Documentation#assembly-usage
```

```
54
55 Different versions of Solidity are used:
56         - Version used: ['=0.5.16', '>=0.5.0']
57         - =0.5.16 (contracts/core/UniswapV2ERC20.sol#1)
58         - =0.5.16 (contracts/core/UniswapV2Factory.sol#1)
59         - =0.5.16 (contracts/core/UniswapV2Pair.sol#1)
60         - >=0.5.0 (contracts/core/interfaces/IERC20.sol#1)
61         - >=0.5.0 (contracts/core/interfaces/IUniswapV2Callee.sol
↳ #1)
62         - >=0.5.0 (contracts/core/interfaces/IUniswapV2ERC20.sol
↳ #1)
63         - >=0.5.0 (contracts/core/interfaces/IUniswapV2Factory.sol
↳ #1)
64         - >=0.5.0 (contracts/core/interfaces/IUniswapV2Pair.sol#1)
65         - =0.5.16 (contracts/core/libraries/Math.sol#1)
66         - =0.5.16 (contracts/core/libraries/SafeMath.sol#1)
67         - =0.5.16 (contracts/core/libraries/UQ112x112.sol#1)
68 Reference: https://github.com/crytic/slither/wiki/Detector-
↳ Documentation#different-pragma-directives-are-used
69
70
71 Pragma version>=0.5.0 (contracts/core/interfaces/IUniswapV2Factory
↳ .sol#1) allows old versions
72 Reference: https://github.com/crytic/slither/wiki/Detector-
↳ Documentation#incorrect-versions-of-solidity
73
74 Parameter UniswapV2Factory.setFeeTo(address)._feeTo (contracts/
↳ core/UniswapV2Factory.sol#78) is not in mixedCase
75 Parameter UniswapV2Factory.setFeeToSetter(address)._feeToSetter (
↳ contracts/core/UniswapV2Factory.sol#83) is not in mixedCase
76 Reference: https://github.com/crytic/slither/wiki/Detector-
↳ Documentation#conformance-to-solidity-naming-conventions
77
78 Variable UniswapV2Factory.createPair(address,address).token0Symbol
↳  (contracts/core/UniswapV2Factory.sol#56) is too similar to
↳ UniswapV2Factory.createPair(address,address).token1Symbol (contra
79 cts/core/UniswapV2Factory.sol#60)
80 Variable UniswapV2Factory.createPair(address,address).
↳ token0Decimals (contracts/core/UniswapV2Factory.sol#58) is too
↳ similar to UniswapV2Factory.createPair(address,address).
↳ token1Decimals (co
81 ntracts/core/UniswapV2Factory.sol#62)
82 Reference: https://github.com/crytic/slither/wiki/Detector-
↳ Documentation#variable-names-are-too-similar
```

```
83
84 UniswapV2Factory.createPair(address,address) (contracts/core/
↳ UniswapV2Factory.sol#41-76) uses literals with too many digits:
85         - bytecode = type(address)(UniswapV2Pair).creationCode (
↳ contracts/core/UniswapV2Factory.sol#46)
86 Reference: https://github.com/crytic/slither/wiki/Detector-
↳ Documentation#too-many-digits
87 contracts/core/UniswapV2Factory.sol analyzed (11 contracts with 78
↳  detectors), 44 result(s) found
```

contracts/core/UniswapV2Pair.sol:

Listing 9

```
1 UniswapV2Pair._update(uint256,uint256,uint112,uint112) (contracts/
↳ core/UniswapV2Pair.sol#96-109) uses a weak PRNG: "blockTimestamp =
↳  uint32(block.timestamp % 2 ** 32) (contracts/core/UniswapV2Pair.
↳ sol#98)"
2 Reference: https://github.com/crytic/slither/wiki/Detector-
↳ Documentation#weak-PRNG
3
4 UniswapV2Pair._safeTransfer(address,address,uint256) (contracts/
↳ core/UniswapV2Pair.sol#53-57) uses a dangerous strict equality:
5         - require(bool,string)(success && (data.length == 0 || abi
↳ .decode(data,(bool))),UniswapV2: TRANSFER_FAILED) (contracts/core/
↳ UniswapV2Pair.sol#56)
6 UniswapV2Pair.mint(address) (contracts/core/UniswapV2Pair.sol
↳ #133-154) uses a dangerous strict equality:
7         - _totalSupply == 0 (contracts/core/UniswapV2Pair.sol#142)
8 Reference: https://github.com/crytic/slither/wiki/Detector-
↳ Documentation#dangerous-strict-equalities
9
10 Reentrancy in UniswapV2Pair.burn(address) (contracts/core/
↳ UniswapV2Pair.sol#157-179):
11         External calls:
12         - _safeTransfer(_token0,to,amount0) (contracts/core/
↳ UniswapV2Pair.sol#171)
13                 - (success,data) = token.call(abi.
↳ encodeWithSelector(SELECTOR,to,value)) (contracts/core/
↳ UniswapV2Pair.sol#54)
14         - _safeTransfer(_token1,to,amount1) (contracts/core/
↳ UniswapV2Pair.sol#172)
15                 - (success,data) = token.call(abi.
```

```
↳ encodeWithSelector(SELECTOR,to,value)) (contracts/core/
↳ UniswapV2Pair.sol#54)
16          State variables written after the call(s):
17          - _update(balance0,balance1,_reserve0,_reserve1) (
↳ contracts/core/UniswapV2Pair.sol#176)
18                  - blockTimestampLast = blockTimestamp (contracts/
↳ core/UniswapV2Pair.sol#107)
19          - kLast = uint256(reserve0).mul(reserve1) (contracts/core/
↳ UniswapV2Pair.sol#177)
20          - _update(balance0,balance1,_reserve0,_reserve1) (
↳ contracts/core/UniswapV2Pair.sol#176)
21                  - reserve0 = uint112(balance0) (contracts/core/
↳ UniswapV2Pair.sol#105)
22          - _update(balance0,balance1,_reserve0,_reserve1) (
↳ contracts/core/UniswapV2Pair.sol#176)
23                  - reserve1 = uint112(balance1) (contracts/core/
↳ UniswapV2Pair.sol#106)
24 Reentrancy in UniswapV2Pair.swap(uint256,uint256,address,bytes) (
↳ contracts/core/UniswapV2Pair.sol#182-210):
25          External calls:
26          - _safeTransfer(_token0,to,amount0Out) (contracts/core/
↳ UniswapV2Pair.sol#193)
27                  - (success,data) = token.call(abi.
↳ encodeWithSelector(SELECTOR,to,value)) (contracts/core/
↳ UniswapV2Pair.sol#54)
28          - _safeTransfer(_token1,to,amount1Out) (contracts/core/
↳ UniswapV2Pair.sol#194)
29                  - (success,data) = token.call(abi.
↳ encodeWithSelector(SELECTOR,to,value)) (contracts/core/
↳ UniswapV2Pair.sol#54)
30          - IUniswapV2Callee(to).uniswapV2Call(msg.sender,amount0Out
↳ ,amount1Out,data) (contracts/core/UniswapV2Pair.sol#195)
31          State variables written after the call(s):
32          - _update(balance0,balance1,_reserve0,_reserve1) (
↳ contracts/core/UniswapV2Pair.sol#208)
33                  - blockTimestampLast = blockTimestamp (contracts/
↳ core/UniswapV2Pair.sol#107)
34          - _update(balance0,balance1,_reserve0,_reserve1) (
↳ contracts/core/UniswapV2Pair.sol#208)
35                  - reserve0 = uint112(balance0) (contracts/core/
↳ UniswapV2Pair.sol#105)
36          - _update(balance0,balance1,_reserve0,_reserve1) (
↳ contracts/core/UniswapV2Pair.sol#208)
37                  - reserve1 = uint112(balance1) (contracts/core/
```

```
↳ UniswapV2Pair.sol#106)
38 Reference: https://github.com/crytic/slither/wiki/Detector-
↳ Documentation#reentrancy-vulnerabilities-1
39
40 UniswapV2Pair.initialize(address,address)._token0 (contracts/core/
↳ UniswapV2Pair.sol#76) lacks a zero-check on :
41                 - token0 = _token0 (contracts/core/UniswapV2Pair.
↳ sol#78)
42 UniswapV2Pair.initialize(address,address)._token1 (contracts/core/
↳ UniswapV2Pair.sol#76) lacks a zero-check on :
43                 - token1 = _token1 (contracts/core/UniswapV2Pair.
↳ sol#79)
44 Reference: https://github.com/crytic/slither/wiki/Detector-
↳ Documentation#missing-zero-address-validation
45
46 Reentrancy in UniswapV2Pair.burn(address) (contracts/core/
↳ UniswapV2Pair.sol#157-179):
47         External calls:
48         - _safeTransfer(_token0,to,amount0) (contracts/core/
↳ UniswapV2Pair.sol#171)
49                 - (success,data) = token.call(abi.
↳ encodeWithSelector(SELECTOR,to,value)) (contracts/core/
↳ UniswapV2Pair.sol#54)
50         - _safeTransfer(_token1,to,amount1) (contracts/core/
↳ UniswapV2Pair.sol#172)
51                 - (success,data) = token.call(abi.
↳ encodeWithSelector(SELECTOR,to,value)) (contracts/core/
↳ UniswapV2Pair.sol#54)
52         State variables written after the call(s):
53         - _update(balance0,balance1,_reserve0,_reserve1) (
↳ contracts/core/UniswapV2Pair.sol#176)
54                 - price0CumulativeLast += uint256(UQ112x112.encode
↳ (_reserve1).uqdiv(_reserve0)) * timeElapsed (contracts/core/
↳ UniswapV2Pair.sol#102)
55         - _update(balance0,balance1,_reserve0,_reserve1) (
↳ contracts/core/UniswapV2Pair.sol#176)
56                 - price1CumulativeLast += uint256(UQ112x112.encode
↳ (_reserve0).uqdiv(_reserve1)) * timeElapsed (contracts/core/
↳ UniswapV2Pair.sol#103)
57 Reentrancy in UniswapV2Pair.swap(uint256,uint256,address,bytes) (
↳ contracts/core/UniswapV2Pair.sol#182-210):
58         External calls:
59         - _safeTransfer(_token0,to,amount0Out) (contracts/core/
↳ UniswapV2Pair.sol#193)
```

```
60                 - (success,data) = token.call(abi.
↳ encodeWithSelector(SELECTOR,to,value)) (contracts/core/
↳ UniswapV2Pair.sol#54)
61         - _safeTransfer(_token1,to,amount1Out) (contracts/core/
↳ UniswapV2Pair.sol#194)
62                 - (success,data) = token.call(abi.
↳ encodeWithSelector(SELECTOR,to,value)) (contracts/core/
↳ UniswapV2Pair.sol#54)
63         - IUniswapV2Callee(to).uniswapV2Call(msg.sender,amount0Out
↳ ,amount1Out,data) (contracts/core/UniswapV2Pair.sol#195)
64         State variables written after the call(s):
65         - _update(balance0,balance1,_reserve0,_reserve1) (
↳ contracts/core/UniswapV2Pair.sol#208)
66                 - price0CumulativeLast += uint256(UQ112x112.encode
↳ (_reserve1).uqdiv(_reserve0)) * timeElapsed (contracts/core/
↳ UniswapV2Pair.sol#102)
67         - _update(balance0,balance1,_reserve0,_reserve1) (
↳ contracts/core/UniswapV2Pair.sol#208)
68                 - price1CumulativeLast += uint256(UQ112x112.encode
↳ (_reserve0).uqdiv(_reserve1)) * timeElapsed (contracts/core/
↳ UniswapV2Pair.sol#103)
69 Reference: https://github.com/crytic/slither/wiki/Detector-
↳ Documentation#reentrancy-vulnerabilities-2
70
71 Reentrancy in UniswapV2Pair.burn(address) (contracts/core/
↳ UniswapV2Pair.sol#157-179):
72         External calls:
73         - _safeTransfer(_token0,to,amount0) (contracts/core/
↳ UniswapV2Pair.sol#171)
74                 - (success,data) = token.call(abi.
↳ encodeWithSelector(SELECTOR,to,value)) (contracts/core/
↳ UniswapV2Pair.sol#54)
75         - _safeTransfer(_token1,to,amount1) (contracts/core/
↳ UniswapV2Pair.sol#172)
76                 - (success,data) = token.call(abi.
↳ encodeWithSelector(SELECTOR,to,value)) (contracts/core/
↳ UniswapV2Pair.sol#54)
77         Event emitted after the call(s):
78         - Burn(msg.sender,amount0,amount1,to,liquidity) (contracts
↳ /core/UniswapV2Pair.sol#178)
79         - Sync(reserve0,reserve1,totalSupply) (contracts/core/
↳ UniswapV2Pair.sol#108)
80                 - _update(balance0,balance1,_reserve0,_reserve1) (
↳ contracts/core/UniswapV2Pair.sol#176)
```

```
Reentrancy in UniswapV2Pair.swap(uint256,uint256,address,bytes) (contracts/core/UniswapV2Pair.sol#182-210):
        External calls:
        - _safeTransfer(_token0,to,amount0Out) (contracts/core/UniswapV2Pair.sol#193)
                - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (contracts/core/UniswapV2Pair.sol#54)
        - _safeTransfer(_token1,to,amount1Out) (contracts/core/UniswapV2Pair.sol#194)
                - (success,data) = token.call(abi.encodeWithSelector(SELECTOR,to,value)) (contracts/core/UniswapV2Pair.sol#54)
        - IUniswapV2Callee(to).uniswapV2Call(msg.sender,amount0Out,amount1Out,data) (contracts/core/UniswapV2Pair.sol#195)
        Event emitted after the call(s):
        - Swap(msg.sender,amount0In,amount1In,amount0Out,amount1Out,to) (contracts/core/UniswapV2Pair.sol#209)
        - Sync(reserve0,reserve1,totalSupply) (contracts/core/UniswapV2Pair.sol#108)
                - _update(balance0,balance1,_reserve0,_reserve1) (contracts/core/UniswapV2Pair.sol#208)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

Different versions of Solidity are used:
        - Version used: ['=0.5.16', '>=0.5.0']
        - =0.5.16 (contracts/core/UniswapV2ERC20.sol#1)
        - =0.5.16 (contracts/core/UniswapV2Pair.sol#1)
        - >=0.5.0 (contracts/core/interfaces/IERC20.sol#1)
        - >=0.5.0 (contracts/core/interfaces/IUniswapV2Callee.sol#1)
        - >=0.5.0 (contracts/core/interfaces/IUniswapV2ERC20.sol#1)
        - >=0.5.0 (contracts/core/interfaces/IUniswapV2Factory.sol#1)
        - >=0.5.0 (contracts/core/interfaces/IUniswapV2Pair.sol#1)
        - =0.5.16 (contracts/core/libraries/Math.sol#1)
        - =0.5.16 (contracts/core/libraries/SafeMath.sol#1)
        - =0.5.16 (contracts/core/libraries/UQ112x112.sol#1)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

Pragma version>=0.5.0 (contracts/core/interfaces/IUniswapV2Pair.
```

```
 ↳ sol#1) allows old versions
109 Reference: https://github.com/crytic/slither/wiki/Detector-
 ↳ Documentation#incorrect-versions-of-solidity
110
111 Low level call in UniswapV2Pair._safeTransfer(address,address,
 ↳ uint256) (contracts/core/UniswapV2Pair.sol#53-57):
112         - (success,data) = token.call(abi.encodeWithSelector(
 ↳ SELECTOR,to,value)) (contracts/core/UniswapV2Pair.sol#54)
113 Low level call in UniswapV2Pair.optimisticAssociation(address) (
 ↳ contracts/core/UniswapV2Pair.sol#86-93):
114         - (success,result) = address(0x167).call(abi.
 ↳ encodeWithSignature(associateToken(address,address),address(this),
 ↳ token)) (contracts/core/UniswapV2Pair.sol#87-88)
115 Reference: https://github.com/crytic/slither/wiki/Detector-
 ↳ Documentation#low-level-calls
116
117 Parameter UniswapV2Pair.initialize(address,address)._token0 (
 ↳ contracts/core/UniswapV2Pair.sol#76) is not in mixedCase
118 Parameter UniswapV2Pair.initialize(address,address)._token1 (
 ↳ contracts/core/UniswapV2Pair.sol#76) is not in mixedCase
119 Function IUniswapV2Pair.DOMAIN_SEPARATOR() (contracts/core/
 ↳ interfaces/IUniswapV2Pair.sol#26) is not in mixedCase
120 Function IUniswapV2Pair.PERMIT_TYPEHASH() (contracts/core/
 ↳ interfaces/IUniswapV2Pair.sol#27) is not in mixedCase
121 Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (contracts/core/
 ↳ interfaces/IUniswapV2Pair.sol#45) is not in mixedCase
122 Reference: https://github.com/crytic/slither/wiki/Detector-
 ↳ Documentation#conformance-to-solidity-naming-conventions
123
124 Variable UniswapV2Pair.swap(uint256,uint256,address,bytes).
 ↳ balance0Adjusted (contracts/core/UniswapV2Pair.sol#203) is too
 ↳ similar to UniswapV2Pair.swap(uint256,uint256,address,bytes).
 ↳ balance1
125 Adjusted (contracts/core/UniswapV2Pair.sol#204)
126 Variable UniswapV2Pair.price0CumulativeLast (contracts/core/
 ↳ UniswapV2Pair.sol#35) is too similar to UniswapV2Pair.
 ↳ price1CumulativeLast (contracts/core/UniswapV2Pair.sol#36)
127 Reference: https://github.com/crytic/slither/wiki/Detector-
 ↳ Documentation#variable-names-are-too-similar
128 contracts/core/UniswapV2Pair.sol analyzed (10 contracts with 78
 ↳ detectors), 32 result(s) found
```

contracts/periphery/UniswapV2Router02.sol:

Listing 10

```
UniswapV2Router02.removeLiquidity(address,address,uint256,uint256,
↳ uint256,address,uint256) (contracts/periphery/UniswapV2Router02.
↳ sol#116-132) ignores return value by IUniswapV2Pair(pair).
↳ transferFrom(msg.sender,pair,liquidity) (contracts/periphery/
↳ UniswapV2Router02.sol#126)
Reference: https://github.com/crytic/slither/wiki/Detector-
↳ Documentation#unchecked-transfer

UniswapV2Router02._swap(uint256[],address[],address).i (contracts/
↳ periphery/UniswapV2Router02.sol#191) is a local variable never
↳ initialized
UniswapV2Library.getAmountsOut(address,uint256,address[]).i (
↳ contracts/periphery/libraries/UniswapV2Library.sol#65) is a local
↳ variable never initialized
UniswapV2Router02._swapSupportingFeeOnTransferTokens(address[],
↳ address).i (contracts/periphery/UniswapV2Router02.sol#300) is a
↳ local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-
↳ Documentation#uninitialized-local-variables

UniswapV2Router02._addLiquidity(address,address,uint256,uint256,
↳ uint256,uint256) (contracts/periphery/UniswapV2Router02.sol#45-72)
↳  ignores return value by IUniswapV2Factory(factory).createPair(
↳ tokenA,tokenB) (contracts/periphery/UniswapV2Router02.sol#55)
Reference: https://github.com/crytic/slither/wiki/Detector-
↳ Documentation#unused-return

UniswapV2Router02.constructor(address,address)._factory (contracts
↳ /periphery/UniswapV2Router02.sol#27) lacks a zero-check on :
                - factory = _factory (contracts/periphery/
↳ UniswapV2Router02.sol#28)
UniswapV2Router02.constructor(address,address)._WHBAR (contracts/
↳ periphery/UniswapV2Router02.sol#27) lacks a zero-check on :
                - WHBAR = _WHBAR (contracts/periphery/
↳ UniswapV2Router02.sol#29)
Reference: https://github.com/crytic/slither/wiki/Detector-
↳ Documentation#missing-zero-address-validation

Different versions of Solidity are used:
        - Version used: ['=0.6.6', '>=0.5.0', '>=0.6.0',
↳ '>=0.6.2']
        - >=0.5.0 (contracts/core/interfaces/IUniswapV2Factory.sol
↳ #1)
```

```
21          - >=0.5.0 (contracts/core/interfaces/IUniswapV2Pair.sol#1)
22          - =0.6.6 (contracts/periphery/UniswapV2Router02.sol#1)
23          - >=0.5.0 (contracts/periphery/interfaces/IERC20.sol#1)
24          - >=0.6.2 (contracts/periphery/interfaces/
↳ IUniswapV2Router01.sol#1)
25          - >=0.6.2 (contracts/periphery/interfaces/
↳ IUniswapV2Router02.sol#1)
26          - >=0.5.0 (contracts/periphery/interfaces/IWHBAR.sol#1)
27          - =0.6.6 (contracts/periphery/libraries/SafeMath.sol#1)
28          - >=0.6.0 (contracts/periphery/libraries/TransferHelper.
↳ sol#3)
29          - >=0.5.0 (contracts/periphery/libraries/UniswapV2Library.
↳ sol#1)
30 Reference: https://github.com/crytic/slither/wiki/Detector-
↳ Documentation#different-pragma-directives-are-used
31
32 TransferHelper.safeApprove(address,address,uint256) (contracts/
↳ periphery/libraries/TransferHelper.sol#7-18) is never used and
↳ should be removed
33 Reference: https://github.com/crytic/slither/wiki/Detector-
↳ Documentation#dead-code
34
35 Pragma version=0.6.6 (contracts/periphery/UniswapV2Router02.sol#1)
↳  allows old versions
36 Pragma version>=0.5.0 (contracts/periphery/interfaces/IERC20.sol
↳ #1) allows old versions
37 Pragma version>=0.6.2 (contracts/periphery/interfaces/
↳ IUniswapV2Router01.sol#1) allows old versions
38 Pragma version>=0.6.2 (contracts/periphery/interfaces/
↳ IUniswapV2Router02.sol#1) allows old versions
39 Pragma version>=0.5.0 (contracts/periphery/interfaces/IWHBAR.sol
↳ #1) allows old versions
40 Pragma version=0.6.6 (contracts/periphery/libraries/SafeMath.sol
↳ #1) allows old versions
41 Pragma version>=0.6.0 (contracts/periphery/libraries/
↳ TransferHelper.sol#3) allows old versions
42 Pragma version>=0.5.0 (contracts/periphery/libraries/
↳ UniswapV2Library.sol#1) allows old versions
43 solc-0.6.6 is not recommended for deployment
44 Reference: https://github.com/crytic/slither/wiki/Detector-
↳ Documentation#incorrect-versions-of-solidity
45
46 Low level call in UniswapV2Router02.optimisticAssociation(address)
↳  (contracts/periphery/UniswapV2Router02.sol#427-435):
```

```
47          - (success,result) = address(0x167).call(abi.
↳ encodeWithSignature(associateToken(address,address),address(this),
↳ token)) (contracts/periphery/UniswapV2Router02.sol#428-429)
48 Low level call in UniswapV2Router02.dissociate(address) (contracts
↳ /periphery/UniswapV2Router02.sol#439-445):
49          - (success,result) = address(0x167).call(abi.
↳ encodeWithSignature(dissociateToken(address,address),address(this)
↳ ,token)) (contracts/periphery/UniswapV2Router02.sol#440-441)
50 Low level call in TransferHelper.safeApprove(address,address,
↳ uint256) (contracts/periphery/libraries/TransferHelper.sol#7-18):
51          - (success,data) = token.call(abi.encodeWithSelector(0
↳ x095ea7b3,to,value)) (contracts/periphery/libraries/TransferHelper
↳ .sol#13)
52 Low level call in TransferHelper.safeTransfer(address,address,
↳ uint256) (contracts/periphery/libraries/TransferHelper.sol#20-31):
53          - (success,data) = token.call(abi.encodeWithSelector(0
↳ xa9059cbb,to,value)) (contracts/periphery/libraries/TransferHelper
↳ .sol#26)
54 Low level call in TransferHelper.safeTransferFrom(address,address,
↳ address,uint256) (contracts/periphery/libraries/TransferHelper.sol
↳ #33-45):
55          - (success,data) = token.call(abi.encodeWithSignature(
↳ transferFrom(address,address,uint256),from,to,value)) (contracts/
↳ periphery/libraries/TransferHelper.sol#40)
56 Low level call in TransferHelper.safeTransferETH(address,uint256)
↳ (contracts/periphery/libraries/TransferHelper.sol#47-50):
57          - (success) = to.call{value: value}(new bytes(0)) (
↳ contracts/periphery/libraries/TransferHelper.sol#48)
58 Reference: https://github.com/crytic/slither/wiki/Detector-
↳ Documentation#low-level-calls
59
60 Function IUniswapV2Pair.DOMAIN_SEPARATOR() (contracts/core/
↳ interfaces/IUniswapV2Pair.sol#26) is not in mixedCase
61 Function IUniswapV2Pair.PERMIT_TYPEHASH() (contracts/core/
↳ interfaces/IUniswapV2Pair.sol#27) is not in mixedCase
62 Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (contracts/core/
↳ interfaces/IUniswapV2Pair.sol#45) is not in mixedCase
63 Variable UniswapV2Router02.WHBAR (contracts/periphery/
↳ UniswapV2Router02.sol#20) is not in mixedCase
64 Function IUniswapV2Router01.WHBAR() (contracts/periphery/
↳ interfaces/IUniswapV2Router01.sol#5) is not in mixedCase
65 Reference: https://github.com/crytic/slither/wiki/Detector-
↳ Documentation#conformance-to-solidity-naming-conventions
66
```

```
67 Variable UniswapV2Router02._addLiquidity(address,address,uint256,
↳ uint256,uint256,uint256).amountADesired (contracts/periphery/
↳ UniswapV2Router02.sol#48) is too similar to UniswapV2Router02._ad
68 dLiquidity(address,address,uint256,uint256,uint256,uint256).
↳ amountBDesired (contracts/periphery/UniswapV2Router02.sol#49)
69 Variable UniswapV2Router02._addLiquidity(address,address,uint256,
↳ uint256,uint256,uint256).amountADesired (contracts/periphery/
↳ UniswapV2Router02.sol#48) is too similar to UniswapV2Router02.add
70 Liquidity(address,address,uint256,uint256,uint256,uint256,address,
↳ uint256).amountBDesired (contracts/periphery/UniswapV2Router02.sol
↳ #77)
71 Variable IUniswapV2Router01.addLiquidity(address,address,uint256,
↳ uint256,uint256,uint256,address,uint256).amountADesired (contracts
↳ /periphery/interfaces/IUniswapV2Router01.sol#10) is too simi
72 lar to UniswapV2Router02.addLiquidity(address,address,uint256,
↳ uint256,uint256,uint256,address,uint256).amountBDesired (contracts
↳ /periphery/UniswapV2Router02.sol#77)
73 Variable UniswapV2Router02._addLiquidity(address,address,uint256,
↳ uint256,uint256,uint256).amountADesired (contracts/periphery/
↳ UniswapV2Router02.sol#48) is too similar to IUniswapV2Router01.ad
74 dLiquidity(address,address,uint256,uint256,uint256,uint256,address
↳ ,uint256).amountBDesired (contracts/periphery/interfaces/
↳ IUniswapV2Router01.sol#11)
75 Variable UniswapV2Router02.addLiquidity(address,address,uint256,
↳ uint256,uint256,uint256,address,uint256).amountADesired (contracts
↳ /periphery/UniswapV2Router02.sol#76) is too similar to Uniswa
76 pV2Router02.addLiquidity(address,address,uint256,uint256,uint256,
↳ uint256,address,uint256).amountBDesired (contracts/periphery/
↳ UniswapV2Router02.sol#77)
77 Variable IUniswapV2Router01.addLiquidity(address,address,uint256,
↳ uint256,uint256,uint256,address,uint256).amountADesired (contracts
↳ /periphery/interfaces/IUniswapV2Router01.sol#10) is too simi
78 lar to IUniswapV2Router01.addLiquidity(address,address,uint256,
↳ uint256,uint256,uint256,address,uint256).amountBDesired (contracts
↳ /periphery/interfaces/IUniswapV2Router01.sol#11)
79 Variable UniswapV2Router02.addLiquidity(address,address,uint256,
↳ uint256,uint256,uint256,address,uint256).amountADesired (contracts
↳ /periphery/UniswapV2Router02.sol#76) is too similar to IUnisw
80 apV2Router01.addLiquidity(address,address,uint256,uint256,uint256,
↳ uint256,address,uint256).amountBDesired (contracts/periphery/
↳ interfaces/IUniswapV2Router01.sol#11)
81 Variable IUniswapV2Router01.addLiquidity(address,address,uint256,
↳ uint256,uint256,uint256,address,uint256).amountADesired (contracts
↳ /periphery/interfaces/IUniswapV2Router01.sol#10) is too simi
```

```
lar to UniswapV2Router02._addLiquidity(address,address,uint256,
↳ uint256,uint256,uint256).amountBDesired (contracts/periphery/
↳ UniswapV2Router02.sol#49)
Variable UniswapV2Router02.addLiquidity(address,address,uint256,
↳ uint256,uint256,uint256,address,uint256).amountADesired (contracts
↳ /periphery/UniswapV2Router02.sol#76) is too similar to Uniswa
pV2Router02._addLiquidity(address,address,uint256,uint256,uint256,
↳ uint256).amountBDesired (contracts/periphery/UniswapV2Router02.sol
↳ #49)
Variable UniswapV2Router02._addLiquidity(address,address,uint256,
↳ uint256,uint256,uint256).amountAOptimal (contracts/periphery/
↳ UniswapV2Router02.sol#66) is too similar to UniswapV2Router02._ad
dLiquidity(address,address,uint256,uint256,uint256,uint256).
↳ amountBOptimal (contracts/periphery/UniswapV2Router02.sol#61)
Reference: https://github.com/crytic/slither/wiki/Detector-
↳ Documentation#variable-names-are-too-similar

removeLiquidityETH(address,uint256,uint256,uint256,address,uint256
↳ ) should be declared external:
        - UniswapV2Router02.removeLiquidityETH(address,uint256,
↳ uint256,uint256,address,uint256) (contracts/periphery/
↳ UniswapV2Router02.sol#133-158)
removeLiquidityETHSupportingFeeOnTransferTokens(address,uint256,
↳ uint256,uint256,address,uint256) should be declared external:
        - UniswapV2Router02.
↳ removeLiquidityETHSupportingFeeOnTransferTokens(address,uint256,
↳ uint256,uint256,address,uint256) (contracts/periphery/
↳ UniswapV2Router02.sol#161-186)
quote(uint256,uint256,uint256) should be declared external:
        - UniswapV2Router02.quote(uint256,uint256,uint256) (
↳ contracts/periphery/UniswapV2Router02.sol#381-383)
getAmountOut(uint256,uint256,uint256) should be declared external:
        - UniswapV2Router02.getAmountOut(uint256,uint256,uint256)
↳ (contracts/periphery/UniswapV2Router02.sol#385-393)
getAmountIn(uint256,uint256,uint256) should be declared external:
        - UniswapV2Router02.getAmountIn(uint256,uint256,uint256) (
↳ contracts/periphery/UniswapV2Router02.sol#395-403)
getAmountsOut(uint256,address[]) should be declared external:
        - UniswapV2Router02.getAmountsOut(uint256,address[]) (
↳ contracts/periphery/UniswapV2Router02.sol#405-413)
getAmountsIn(uint256,address[]) should be declared external:
        - UniswapV2Router02.getAmountsIn(uint256,address[]) (
↳ contracts/periphery/UniswapV2Router02.sol#415-423)
balance() should be declared external:
```

```
104         - UniswapV2Router02.balance() (contracts/periphery/
 ↳ UniswapV2Router02.sol#447-449)
105 Reference: https://github.com/crytic/slither/wiki/Detector-
 ↳ Documentation#public-function-that-could-be-declared-external
106 contracts/periphery/UniswapV2Router02.sol analyzed (10 contracts
 ↳ with 78 detectors), 49 result(s) found
```

As a result of the tests carried out with the Slither tool, some results were obtained and reviewed by Halborn. Based on the results reviewed, some vulnerabilities were determined to be false positives.

THANK YOU FOR CHOOSING

// HALBORN