HACKEN

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Customer: SaucerSwap
Date: July 12th, 2022

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

This document may contain confidential information about IT systems and the intellectual property of the Customer as well as information about potential vulnerabilities and methods of their exploitation.

The report containing confidential information can be used internally by the Customer, or it can be disclosed publicly after all vulnerabilities are fixed — upon a decision of the Customer.

Document

Name	Smart Contract Code Review and Security Analysis Report for SaucerSwap
Approved By	Evgeniy Bezuglyi | SC Audits Department Head at Hacken OU
Type	HTS token; Staking
Platform	Hedera Network
Language	Solidity
Methods	Manual Review, Automated Review, Architecture review
Website	saucerswap.finance
Timeline	07.06.2022 - 12.07.2022
Changelog	23.06.2022 - Initial Review 12.07.2022 - Second Review

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Table of contents

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Introduction

Hacken OÜ (Consultant) was contracted by SaucerSwap (Customer) to conduct a Smart Contract Code Review and Security Analysis. This report presents the findings of the security assessment of the Customer's smart contracts.

Scope

The scope of the project is smart contracts in the repository:

Initial review scope
Repository:
https://github.com/vea-vecturne/saucerswap-core
Commit:
bd4029
Technical Documentation:

Type: Technical description
Link

Integration and Unit Tests: Yes
Deployed Contracts Addresses: -
Contracts:
File: ./contracts/hedera/HederaResponseCodes.sol
SHA3:
b97c19959a7cc7d63470cf66f7768b445624e50e8b38b638fb984537a98d74b8

File: ./contracts/hedera/HederaTokenService.sol
SHA3:
b8f8e91d224bd711bc2e543ba4f64c19726bc45946cc352960ddfd63148e31cf

File: ./contracts/hedera/IExchangeRate.sol
SHA3:
653b75ab25c7cc7103a8f09f14bfa8b6e0b1dccce713757757636a38ae295798

File: ./contracts/hedera/IHederaTokenService.sol
SHA3:
5af22ffe70d2967080cb4f27bb1d3b370fe0da000f076f36ff84a9d9c08980f2

File: ./contracts/hedera/SafeHederaTokenService.sol
SHA3:
b3dfbab0606fe79134d80ac6f5185ab130462549ef10e9763f959936ab46f66d

File: ./contracts/interfaces/IERC20.sol
SHA3:
6707751db33963ade868182e1673bfddf838e034d54c208f92484211976f2fdb

File: ./contracts/interfaces/IUniswapV2Callee.sol
SHA3:
370ed4f22ad745de00c6f784b0c9a2c944a830922635f6a9bab90bffb8af9ea6

File: ./contracts/interfaces/IUniswapV2Factory.sol
SHA3:
58760319b1bd030d5b6fd56b7a0fe18a9035926ef44f953edad92688b26074e5

```
      File: ./contracts/interfaces/IUniswapV2Pair.sol
      SHA3:
d6b96ee7b15b7f4631dc966c072fcdbc27dfe535c2a17e904fcd9b1b3d4807aa

      File: ./contracts/interfaces/IUniswapV2Router01.sol
      SHA3:
fbd1ba9a40f1effa00bfa668cffc0bd148aa349e96a45ea5f7681623f562302d

      File: ./contracts/interfaces/IUniswapV2Router02.sol
      SHA3:
610470051aa8eee3c8fcead675d220a40432b2c23bbf608459475d9d9ae2a2ce

      File: ./contracts/interfaces/IWHBAR.sol
      SHA3:
3842bc53b378e524c4b0ef5458be9e44d959ebd6cfc60e3198d5ec5e2bb83ac3

      File: ./contracts/libraries/Bits.sol
      SHA3:
8b5124f0b9ea04aa827c109d789d753588a8f894047cf0da0d46b64b093c7d13

      File: ./contracts/libraries/Math.sol
      SHA3:
663e6a23a9c6451c976d7fecd5ab3a07f489741a19baf0a18de7f9c41b39bd80

      File: ./contracts/libraries/SafeCast.sol
      SHA3:
b769c75b54cd951eb036150135b5c7760f22a9d9dc5c2586b35e7e211a7e9374

      File: ./contracts/libraries/SafeMath.sol
      SHA3:
fab420417658e540599c9a866ccc05034ffddfc5e0f2e619704c3175cad627a6

      File: ./contracts/libraries/TransferHelper.sol
      SHA3:
2f15e5c451b0d06805d2dc07746749ba2fc85993851b04c0bb99248854876ea8

      File: ./contracts/libraries/UniswapV2Library.sol
      SHA3:
beed609084ed81bb585bca978a625b024b5620b0a864d10f123adf334d5479ee

      File: ./contracts/libraries/UQ112x112.sol
      SHA3:
1bd6aab05ce80428653714c620157c83d66b2344f23faba98ad0b958ab2b2a46

      File: ./contracts/Migrations.sol
      SHA3:
abdc29f1b06e929fbe396edaeeee6e0c33e57a61602ffe06ca6aef9a791e5d7b

      File: ./contracts/multicall/Multicall.sol
      SHA3:
86fa8a682f976bda26ae1b5124e71fd31809d46ff8aae9219819f17cfb455973

      File: ./contracts/UniswapV2Factory.sol
```

SHA3:
faacb8dad3ff8e92720e03d17a4f1ca2e8f9648b51dae2b08a45d21bc538d167

File: ./contracts/UniswapV2Pair.sol
SHA3:
4cbddd3c6a37bbc8104f53d41045e778d9232d36000100d6fbf57d71682f67e2

File: ./contracts/UniswapV2Router02.sol
SHA3:
ce037861f1652c3cead192368d612747d518e1251a57c834e5711c67a2e6a55e

File: ./contracts/WHBAR.sol
SHA3: aff20c4ad6daf56424cf3f78148cbcd5dd8a3ff0be8cb1f9b64cfcf777df1b85

Second review scope

Repository:
https://github.com/vea-vecturne/saucerswap-core
Commit:
db2eab3a8d563622cdac381a0a475c300755b1bd
Technical Documentation:

Type: Technical description
Link

Type: Functional Requirements
Link
Integration and Unit Tests: No
Deployed Contracts Addresses: No
Contracts:
File: ./contracts/hedera/HederaResponseCodes.sol
SHA3:
f46f133bc0c30e7eb8490712de110516267c14406ae2212d6d9bfcd8479c6da8

File: ./contracts/hedera/HederaTokenService.sol
SHA3:
cfa431b405a5558fd563648033dbc38a7608d622837f4e158fea84af7072d1c8

File: ./contracts/hedera/IExchangeRate.sol
SHA3:
1ba4ea39073c951727b36d47fbff3f8f617cdbb33167606ebc622f0b0c9a0cb5

File: ./contracts/hedera/IHederaTokenService.sol
SHA3:
dbbd795afced4cfcb221638b0591a9dd436add078d97b8ce94bf57d80901c28a

File: ./contracts/hedera/SafeHederaTokenService.sol
SHA3:
bea372e8d946c36603e4d45cef11d1c88d424bd9ea3e91622b4793e3b7d2291c

File: ./contracts/interfaces/IERC20.sol
SHA3:
b8b455cadd050f4d686499059d0fc41aac92c228344984dbb278fffba84c1969

File: ./contracts/interfaces/IUniswapV2Callee.sol

```
      SHA3:
2b3248660e6839ab16804ee2c249b16feb113afdcce1cdba2a7257e04eaf5742

      File: ./contracts/interfaces/IUniswapV2Factory.sol
      SHA3:
1d330708849026e72d8209251a0cf771c28392413f204b1be40cdf14076462d8

      File: ./contracts/interfaces/IUniswapV2Pair.sol
      SHA3:
8e9a83b36903b1145ccbbff7ddd64ea842140742917a79630f7684c5848e04a3

      File: ./contracts/interfaces/IUniswapV2Router01.sol
      SHA3:
8ab1371ca493599b804ce3bfb91dfa37c98b9ce49951f524b4c10de9fc7e2a5d

      File: ./contracts/interfaces/IUniswapV2Router02.sol
      SHA3:
1e6f8bd5a7fa470f14bbe7801dc9b992d607d998d4ce34c4e6c792bee43b12d0

      File: ./contracts/interfaces/IWHBAR.sol
      SHA3:
ba9c4a8ab09cb61d0a4bc9a94146490a6ebd0bb0e0aa9830f3be37ab958a207c

      File: ./contracts/libraries/Bits.sol
      SHA3:
8b5124f0b9ea04aa827c109d789d753588a8f894047cf0da0d46b64b093c7d13

      File: ./contracts/libraries/Math.sol
      SHA3:
663e6a23a9c6451c976d7fecd5ab3a07f489741a19baf0a18de7f9c41b39bd80

      File: ./contracts/libraries/SafeCast.sol
      SHA3:
b769c75b54cd951eb036150135b5c7760f22a9d9dc5c2586b35e7e211a7e9374

      File: ./contracts/libraries/SafeMath.sol
      SHA3:
fab420417658e540599c9a866ccc05034ffddfc5e0f2e619704c3175cad627a6

      File: ./contracts/libraries/TransferHelper.sol
      SHA3:
9ae957aa2c97fded9780f3dcf6c00b2c01b9cc0ce50de30a47b7f5f2bde8b2ec

      File: ./contracts/libraries/UniswapV2Library.sol
      SHA3:
dc23bdc8eab36112d8f90e500efd28bc540e8149a58e6a3f8bb62bd08695dc87

      File: ./contracts/libraries/UQ112x112.sol
      SHA3:
1bd6aab05ce80428653714c620157c83d66b2344f23faba98ad0b958ab2b2a46

      File: ./contracts/Migrations.sol
      SHA3:
abdc29f1b06e929fbe396edaeeee6e0c33e57a61602ffe06ca6aef9a791e5d7b
```

```
      File: ./contracts/UniswapV2Factory.sol
      SHA3:
baec90cb65564ddf448d00f635f73c449bace876362cd4741f0cfbd60cb8fb1c

      File: ./contracts/UniswapV2Pair.sol
      SHA3:
1fed6efbeab967115c4d91f1b7663b65548368e9c4c537444adfd8aefa8b049f

      File: ./contracts/UniswapV2Router02.sol
      SHA3:
a74bd9b525955d4e16d8faa45772d994e93809e6785484aada3746bc7908d5e5

      File: ./contracts/WHBAR.sol
      SHA3:
acadb1124b276c537375a29e06b33aa6fb81e5faa13236b53d0e7fc68181153e
```

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Severity Definitions

Risk Level	Description
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to assets loss or data manipulations.
High	High-level vulnerabilities are difficult to exploit; however, they also have a significant impact on smart contract execution, e.g., public access to crucial functions.
Medium	Medium-level vulnerabilities are important to fix; however, they cannot lead to assets loss or data manipulations.
Low	Low-level vulnerabilities are mostly related to outdated, unused, etc. code snippets that cannot have a significant impact on execution.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Executive Summary

The score measurement details can be found in the corresponding section of the methodology.

Documentation quality

The total Documentation Quality score is **10** out of **10**. The Customer provided technical and functional requirements.

Code quality

The total CodeQuality score is **7** out of **10**. Code is not fully following the official Style Guide. An unused variable is detected.

All found issues are displayed in the “Findings” section.

Architecture quality

The architecture quality score is **7** out of **10**. Development environment with README was provided. Tests were provided, but they were not running.

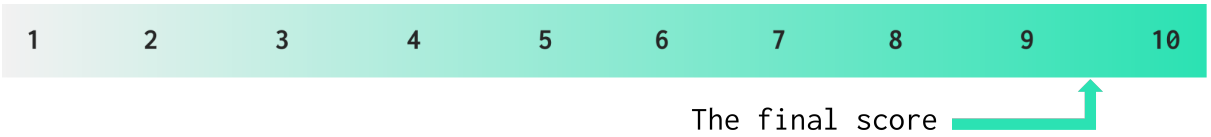
Security score

As a result of the audit, the code contains **4** low severity issues. The security score is **10** out of **10**.

All found issues are displayed in the “Findings” section.

Summary

According to the assessment, the Customer's smart contract has the following score: **9.4**.



Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Checked Items

We have audited provided smart contracts for commonly known and more specific vulnerabilities. Here are some of the items that are considered:

Item	Type	Description	Status
Default Visibility	SWC-100 SWC-108	Functions and state variables visibility should be set explicitly. Visibility levels should be specified consciously.	Passed
Integer Overflow and Underflow	SWC-101	If unchecked math is used, all math operations should be safe from overflows and underflows.	Passed
Outdated Compiler Version	SWC-102	It is recommended to use a recent version of the Solidity compiler.	Failed
Floating Pragma	SWC-103	Contracts should be deployed with the same compiler version and flags that they have been tested thoroughly.	Passed
Unchecked Call Return Value	SWC-104	The return value of a message call should be checked.	Passed
Access Control & Authorization	CWE-284	Ownership takeover should not be possible. All crucial functions should be protected. Users could not affect data that belongs to other users.	Passed
SELFDESTRUCT Instruction	SWC-106	The contract should not be self-destructible while it has funds belonging to users.	Passed
Check-Effect-Interaction	SWC-107	Check-Effect-Interaction pattern should be followed if the code performs ANY external call.	Passed
Uninitialized Storage Pointer	SWC-109	Storage type should be set explicitly if the compiler version is < 0.5.0.	Not Relevant
Assert Violation	SWC-110	Properly functioning code should never reach a failing assert statement.	Not Relevant
Deprecated Solidity Functions	SWC-111	Deprecated built-in functions should never be used.	Passed
Delegatecall to Untrusted Callee	SWC-112	Delegatecalls should only be allowed to trusted addresses.	Passed
DoS (Denial of Service)	SWC-113 SWC-128	Execution of the code should never be blocked by a specific contract state unless it is required.	Passed

Race Conditions	SWC-114	Race Conditions and Transactions Order Dependency should not be possible.	Passed
Authorization through tx.origin	SWC-115	tx.origin should not be used for authorization.	Passed
Block values as a proxy for time	SWC-116	Block numbers should not be used for time calculations.	Passed
Signature Unique Id	SWC-117 SWC-121 SWC-122	Signed messages should always have a unique id. A transaction hash should not be used as a unique id.	Passed
Shadowing State Variable	SWC-119	State variables should not be shadowed.	Passed
Weak Sources of Randomness	SWC-120	Random values should never be generated from Chain Attributes or be predictable.	Passed
Incorrect Inheritance Order	SWC-125	When inheriting multiple contracts, especially if they have identical functions, a developer should carefully specify inheritance in the correct order.	Passed
Calls Only to Trusted Addresses	EEA-Level-2 SWC-126	All external calls should be performed only to trusted addresses.	Passed
Presence of unused variables	SWC-131	The code should not contain unused variables if this is not justified by design.	Failed
EIP standards violation	EIP	EIP standards should not be violated.	Not Relevant
Assets integrity	**Custom**	Funds are protected and cannot be withdrawn without proper permissions.	Passed
User Balances manipulation	**Custom**	Contract owners or any other third party should not be able to access funds belonging to users.	Passed
Data Consistency	**Custom**	Smart contract data should be consistent all over the data flow.	Passed
Flashloan Attack	**Custom**	When working with exchange rates, they should be received from a trusted source and not be vulnerable to short-term rate changes that can be achieved by using flash loans. Oracles should be used.	Passed
Token Supply manipulation	**Custom**	Tokens can be minted only according to rules specified in a whitepaper or any other documentation provided by the customer.	Passed
Gas Limit and	**Custom**	Transaction execution costs should not	Passed

Loops		depend dramatically on the amount of data stored on the contract. There should not be any cases when execution fails due to the block Gas limit.	
Style guide violation	**Custom**	Style guides and best practices should be followed.	Failed
Requirements Compliance	**Custom**	The code should be compliant with the requirements provided by the Customer.	Passed
Environment Consistency	**Custom**	The project should contain a configured development environment with a comprehensive description of how to compile, build and deploy the code.	Passed
Tests Coverage	**Custom**	The code should be covered with unit tests. Test coverage should be 100%, with both negative and positive cases covered. Usage of contracts by multiple users should be tested.	Passed
Stable Imports	**Custom**	The code should not reference draft contracts, that may be changed in the future.	Passed

System Overview

Core repo is a fork of Uniswap with the following contracts:

- *HederaTokenService.sol* - Standard Hedera Token Service, which provides common facilities like minting, burning, transferring, and getting exchange rates via pre-compiled contracts.
- *SafeHederaTokenService.sol* - Safe calls decorator for the HederaTokenService.sol
- *Migrations.sol* - Standard Truffle Migrations contract
- UniswapV2Factory.sol - Fork of the Uniswap factory with next changes:
    - Create Hedera Fungible Token upon pair creation.
    - Collect additional fees upon pair creation, including the fee needed to create Hedera token.
- *UniswapV2Pair.sol* - Standard Uniswap pair. The main change is that work with Hedera tokens takes place under the hood.
- *UniswapV2Router02.sol* - Standard Uniswap router with next changes:
    - Adding a liquidity pool does not create a new pair under the hood - this functionality is moved into a separate function.
    - Redundant removal liquidity functions are cut out.
    - Works with WHBAR - fork of the WETH
      - Using transferring functionality of Hedera instead of standard.

Privileged roles

No privileged roles were found.

Risks

Account and entity owners must ensure that linked Auto-Renew and admin accounts have sufficient balances for auto-renewal fees or risk permanent removal of their entity. *UniswapV2Pair* contract, *createFungible* function.

Findings

Critical

No high severity issues were found.

High

1. **Potential DoS**

   If a destination address argument of the *send* method is a contract address and this contract has a fallback or receive function, the transfer will fail due to the 2300 Gas limit. Lines 75, 86.

   Possible DoS.

   File: ./contracts/WHBAR.sol

   Contract: WHBAR

   Function: withdraw

   Recommendation: Ensure that the address argument is an externally-owned address or use a call method instead of send.

   Status: Fixed (Revised commit: db2eab3a8d563622cdac381a0a475c300755b1bd)

Medium

1. **Unchecked return value of a call method**

   Result of the call to contract is not checked on line 65.

   Therefore *rentPayer* may not receive funds in case of a failed transaction.

   File: ./contracts/UniswapV2Factory.sol

   Contract: UniswapV2Factory

   Function: createPair

   Recommendation: Check the return value of a call method and revert transaction in case it is *false* or emit specific even to log it.

   Status: Fixed (Revised commit: db2eab3a8d563622cdac381a0a475c300755b1bd)

2. **Floating pragma**

   The contracts use floating pragma. Moreover, Solidity compiler version 0.9.0 is not yet available.

   Files: ./contracts/libraries/UniswapV2Library.sol, ./contracts/libraries/SafeMath.sol, ./contracts/hedera/HederaResponseCodes.sol, ./contracts/hedera/HederaTokenService.sol, ./contracts/hedera/IExchangeRate.sol,

./contracts/hedera/IHederaTokenService.sol,
./contracts/interfaces/IERC20.sol,
./contracts/interfaces/IUniswapV2Callee.sol,
./contracts/interfaces/IUniswapV2Factory.sol,
./contracts/interfaces/IUniswapV2Pair.sol,
./contracts/interfaces/IUniswapV2Router01.sol,
./contracts/interfaces/IUniswapV2Router02.sol,
./contracts/interfaces/IWHBAR.sol

Contracts: UniswapV2Library.sol, SafeMath.sol, HederaResponseCodes.sol, HederaTokenService.sol, IExchangeRate.sol, IHederaTokenService.sol, IERC20.sol, IUniswapV2Callee.sol, IUniswapV2Factory.sol, IUniswapV2Pair.sol, IUniswapV2Router01.sol, IUniswapV2Router02.sol, IWHBAR.sol.

Recommendation: Consider locking the pragma version whenever possible and avoid using a floating pragma in the final deployment.

Status: Fixed (Revised commit: db2eab3a8d563622cdac381a0a475c300755b1bd)

3. Return variable type inconsistency

The *responseCode* return parameter is defined as *int(as default int256)* in the specified functions. Whereas the result of the call to the pre-compiled contract is marked as int32.

Using different types for the same variable can cause data not to fit.

File: ./contracts/hedera/HederaTokenService.sol

Contract: HederaTokenService

Functions: createFungibleToken, mintToken, burnToken, associateTokens, associateToken, transferToken, transferTokenRouter, SafeHederaTokenService.safeMintToken,
SafeHederaTokenService.safeBurnToken,
SafeHederaTokenService.safeAssociateTokens,
SafeHederaTokenService.safeAssociateToken,
SafeHederaTokenService.safe

Recommendation: Review and use singular type.

Status: Fixed (Revised commit: db2eab3a8d563622cdac381a0a475c300755b1bd)

4. Confusing function name

The function has the same signature as ERC-20 *balanceOf* function. However, it is used for other purposes.

Users of the Pair contract can misunderstand the purpose of the function and implement their code considering the Pair contract ERC-20 compliant.

File: ./contracts/UniswapV2Pair.sol

Contract: UniswapV2Pair

Function: balanceOf

Recommendation: Rename the function.

Status: Fixed (Revised commit: db2eab3a8d563622cdac381a0a475c300755b1bd)

Low

1. Outdated compiler version

Using an outdated compiler version can be problematic, especially if publicly disclosed bugs and issues affect the current compiler version.

Files: all

Contracts: all

Recommendation: Use a recent version of the Solidity compiler, at least 0.8.0

Status: Reported (Revised commit: db2eab3a8d563622cdac381a0a475c300755b1bd)

2. Style guide violation

Contracts do not follow the Solidity code style guide.

Files: ./contracts/UniswapV2Pair.sol, ./contracts/UniswapV2Router02.sol

Contracts: UniswapV2Pair.sol, UniswapV2Router02.sol,

Recommendation: Follow the official Solidity code style guide. Pay attention to the next topics: “Order of Functions“, “Line length“.

Status: Reported (Revised commit: db2eab3a8d563622cdac381a0a475c300755b1bd)

3. Unused contract

Contract *Multicall* is defined in the audit scope but never used; this makes the scope overwhelmed.

File: ./contracts/multicall/Multicall.sol

Contract: Multicall.sol

Recommendation: Remove this contract.

Status: Fixed (Revised commit: db2eab3a8d563622cdac381a0a475c300755b1bd)

4. **Revert transaction message is not specified**

   Revert transaction reason is not specified as a second argument of the *require* function on lines 76, 87.

   Therefore in case of a transaction revert, it will be harder to debug the contract without explanation.

   File: ./contracts/WHBAR.sol

   Contract: WHBAR.sol

   Function: withdraw

   Recommendation: Provide a clear reason for the *require* statement.

   Status: Fixed (Revised commit: db2eab3a8d563622cdac381a0a475c300755b1bd)

5. **Revert transaction message is not specified**

   Revert transaction reason is not specified as a second argument of the *require* function.

   Therefore in case of a transaction revert, it will be harder to debug the contract without explanation.

   File: ./contracts/hedera/HederaTokenService.sol

   Contract: HederaTokenService.sol

   Functions: tinycentsToTinybars, tinybarsToTinycents

   Recommendation: Provide a clear reason for the *require* statement.

   Status: Fixed (Revised commit: db2eab3a8d563622cdac381a0a475c300755b1bd)

6. **Unused variables**

   Variables *TINY_PARTS_PER_WHOLE*, *ADMIN_KEY_TYPE*, *KYC_KEY_TYPE*, *FREEZE_KEY_TYPE*, *WIPE_KEY_TYPE*, *SUPPLY_KEY_TYPE*, *FEE_SCHEDULE_KEY_TYPE*, *PAUSE_KEY_TYPE*, *INIT_CODE_PAIR_HASH* are declared but never used.

   Unused variables make code overwhelmed and decrease readability.

   Files: ./contracts/hedera/HederaTokenService.sol, ./contracts/UniswapV2Factory.sol

   Contracts: HederaTokenService, UniswapV2Factory

   Functions: -

   Recommendation: Remove unused variables.

Status: Fixed (Revised commit: db2eab3a8d563622cdac381a0a475c300755b1bd)

7. **Deprecated pragma experimental declaration**

   The pragma `*pragma experimental ABIEncoderV2;*` is valid but deprecated and has no effect from version 0.8.0.

   Files: ./contracts/UniswapV2Pair.sol, ./contracts/WHBAR.sol, ./contracts/interfaces/IUniswapV2Router02.sol, ./contracts/hedera/SafeHederaTokenService.sol, ./contracts/multicall/Multicall.sol, ./contracts/hedera/HederaTokenService.sol, ./contracts/hedera/IHederaTokenService.sol

   Contracts: UniswapV2Pair.sol, WHBAR.sol, IUniswapV2Router02.sol, SafeHederaTokenService.sol, Multicall.sol, HederaTokenService.sol, IHederaTokenService.sol,

   Recommendation: Use pragma abicoder v2; instead to be explicit.

   Status: Reported (Revised commit: db2eab3a8d563622cdac381a0a475c300755b1bd)

8. **Unused method**

   tokenCreateFee is defined but never used.

   File: ./contracts/interfaces/IUniswapV2Factory.sol

   Contract: IUniswapV2Factory.sol

   Function: tokenCreateFee

   Recommendation: Remove redundant method.

   Status: Fixed (Revised commit: db2eab3a8d563622cdac381a0a475c300755b1bd)

9. **Unused method**

   kLast is defined but never used.

   File: ./contracts/interfaces/IUniswapV2Pair.sol

   Contract: IUniswapV2Pair.sol

   Function: kLast

   Recommendation: Remove redundant method.

   Status: Fixed (Revised commit: db2eab3a8d563622cdac381a0a475c300755b1bd)

10. **Validation should be on top of the function**

    In order to save Gas purposes, the *required* statement which validates whether a pair exists should be on top of the function before

calculating the amount code. It is no sense to run amounts calculation code if the pair does not exist.

File: ./contracts/UniswapV2Router02.sol

Contract: UniswapV2Router02.sol

Function: addLiquidityETH

Recommendation: Move the *required* statement on top of the function.

Status: Fixed (Revised commit: db2eab3a8d563622cdac381a0a475c300755b1bd)

11. Hardcoded value

The HederaResponseCodes contains code for successful execution. Though the hardcoded value is used on L#272.

File: ./contracts/UniswapV2Pair.sol

Contract: UniswapV2Pair.sol

Function: createFungible

Recommendation: Use constant instead.

Status: Fixed (Revised commit: db2eab3a8d563622cdac381a0a475c300755b1bd)

12. Unused variables

Variable *INIT_CODE_PAIR_HASH* is declared but never used.

Unused variables make code overwhelmed and decrease readability.

Files: ./contracts/hedera/HederaTokenService.sol, ./contracts/UniswapV2Factory.sol

Contract: UniswapV2Factory

Functions: -

Recommendation: Remove the unused variable.

Status: New

Disclaimers

Hacken Disclaimer

The smart contracts given for audit have been analyzed by the best industry practices at the date of this report, with cybersecurity vulnerabilities and issues in smart contract source code, the details of which are disclosed in this report (Source Code); the Source Code compilation, deployment, and functionality (performing the intended functions).

The audit makes no statements or warranties on the security of the code. It also cannot be considered a sufficient assessment regarding the utility and safety of the code, bug-free status, or any other contract statements. While we have done our best in conducting the analysis and producing this report, it is important to note that you should not rely on this report only – we recommend proceeding with several independent audits and a public bug bounty program to ensure the security of smart contracts.

Technical Disclaimer

Smart contracts are deployed and executed on a blockchain platform. The platform, its programming language, and other software related to the smart contract can have vulnerabilities that can lead to hacks. Thus, the audit cannot guarantee the explicit security of the audited smart contracts.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io