verichains

SECURITY AUDIT OF

RABBITSWAP V3 SMART CONTRACTS

Public Report

Nov 15, 2024

Verichains Lab

info@verichains.io

https://www.verichains.io

Driving Technology > Forward

ABBREVIATIONS

Name	Description
Ethereum	An open source platform based on blockchain technology to create and distribute smart contracts and decentralized applications.
Ether (ETH)	A cryptocurrency whose blockchain is generated by the Ethereum platform. Ether is used for payment of transactions and computing services in the Ethereum network.
Smart contract	A computer protocol intended to digitally facilitate, verify or enforce the negotiation or performance of a contract.
Solidity	A contract-oriented, high-level language for implementing smart contracts for the Ethereum platform.
Solc	A compiler for Solidity.
ERC20	ERC20 (BEP20 in Binance Smart Chain or *x*RP20 in other chains) tokens are blockchain-based assets that have value and can be sent and received. The primary difference with the primary coin is that instead of running on their own blockchain, ERC20 tokens are issued on a network that supports smart contracts such as Ethereum or Binance Smart Chain.
ERC721	The ERC-721 introduces a standard for NFT, in other words, this type of Token is unique and can have different value than another Token from the same Smart Contract, maybe due to its age, rarity or even something else like its visual.

EXECUTIVE SUMMARY

This Security Audit Report was prepared by Verichains Lab on Nov 15, 2024. We would like to thank the RabbitSwap for trusting Verichains Lab in auditing smart contracts. Delivering high-quality audits is always our top priority.

This audit focused on identifying security flaws in code and the design of the RabbitSwap V3 smart contracts. The scope of the audit is limited to the source code files provided to Verichains. Verichains Lab completed the assessment using manual, static, and dynamic analysis techniques.

During the audit process, the audit team had identified some issues in the source code, along with some recommendations.

TABLE OF CONTENTS

1. MANAGEMENT SUMMARY

1.1. About RabbitSwap V3 smart contracts

RabbitSwap is a decentralized exchange (DEX) that is a fork of Uniswap V3, incorporating advanced features to enhance user experience and efficiency.

One of the key innovations in RabbitSwap is the implementation of the ZeroGas mechanic on Viction, which allows users to perform transactions with zero gas fees under certain conditions. This feature aims to reduce the cost of trading and make the platform more accessible to a broader audience.

By leveraging the proven architecture of Uniswap V3 and introducing unique enhancements, RabbitSwap offers a robust and user-friendly platform for decentralized trading.

1.2. Audit Scope

This audit focused on identifying security flaws in code and the design of the RabbitSwap V3 smart contracts. It was conducted on commit `14fea8e1232b89b5abd53e4f85d395bfeec48635` from git repository *https://github.com/RabbitDEX/rabbitv3-contracts*

The RabbitSwap V3 smart contracts are forked from the following sources:

`Uniswap V3 Core`:

- Forked from *https://github.com/Uniswap/v3-core/commit/d8b1c635c275d2a9450bd6a78f3fa2484fef73eb*
- Add support protocol fee up to 10,000 denominator.

`Uniswap V3 Periphery`:

- Forked from *https://github.com/Uniswap/v3-periphery/commit/0682387198a24c7cd63566a2c58398533860a5d1*
- Add support VIC ZeroGas to perform gas-less transaction.

`VRC725`:

- Forked from *https://github.com/BuildOnViction/vrc725/commit/363fd85a4da8454d873c81673bdcb4ae6640f1ac*

`VRC25`:

- Forked from *https://github.com/BuildOnViction/vrc25/commit/5806a24c4f23242a5f0880b8e92c883bea1b98fc*
- Downgrade smart contracts to solidity `v0.7.6`

- Use interfaces and libraries from `OpenZeppelin`
- Implement `IERC721Enumerable`

The latest version of the following file was made available in the course of the review:

SHA256 Sum	File
d22e664f310be395ff85e5c1e62089beda97021d84189db13fc94af9910e6860	./vrc725/VRC725.sol
a761fca47a392560b99c36c46ce41fd3267d2d0beffa6ca3b322ded9168e609b	./vrc725/libraries/ECDSA.sol
ce40be37a3eac073ccd1f419cee1001ffa85eaa1c089ea771951773c91f452dc	./vrc725/libraries/SignatureChecker.sol
b4d035a4e7e1c8902a57c79db77c16b1cce9d279acfc99ba74ad04d162f4f42e	./vrc725/libraries/ERC165.sol
3ed6c9e37f821f7222cada49499451c612bcef42b02a8f397be2f75211d70f5b	./core/RabbitSwapV3Factory.sol
cc3b4baf0fb1329887fa04d5d5666edaa7c552dc606f4700ce6ee394cc9c34fd	./core/libraries/Position.sol
84d20a16d5346f6ec4c12dff4df23dda5d46e52d33f18aaaaac2e9e36ce4a072	./core/libraries/LiquidityMath.sol
bd7a17c5134f0718eb7d856ddfc58d8347d32a8f661bed53aa3ad17c9aea09ba	./core/libraries/TickBitmap.sol
804f435e2c745d8db1ce8a19450dee83f61c98910a24920ae3e20cd439dea725	./core/libraries/Tick.sol
32f71ea9156f55572a72efb0b2a913df88de66ff33d042043fb3e51a6050a557	./core/libraries/BitMath.sol
394107ff2dbbaded5612452af5e77b4af9d0871b096c1514b0ea659b862fc46f	./core/libraries/LowGasSafeMath.sol
a937160b3575fa6d6bc70b5b2723f9433355714a3cf5d65f2dcaa46b505d1654	./core/libraries/Oracle.sol
e68d3c6c8c847d16c287c1f17f684d11cf0f6267a8d8dc29e81656bccf68f062	./core/libraries/TickMath.sol
cfc3aef8851f183492547dccc168bf72398fba2aad4c4d9d4784f542a8ccda34	./core/libraries/FixedPoint128.sol

ddd62e3a94346248677f30f1ab009ef015e71e4b8696dcca890eeabc9dc6c149	./core/libraries/SqrtPriceMath.sol
f15b94bc000ee68e434c3ff4c28696c3f5bc23222bd6249744a7228b063e267b	./core/libraries/TransferHelper.sol
959c52e1c860bfbede3e33c8d04910c11038973173aa3478547cbad2e444880e	./core/libraries/FullMath.sol
9aed494b56d3dd16b7d6535583ded2cdfb03dc80aaa919347b13d35fd597e8bf	./core/libraries/SafeCast.sol
219deb88ffbcdefa482be35051db586378e8523062bee592dd2c5fa7fb47ebd6	./core/libraries/FixedPoint96.sol
4d02353eb503e3111e25bd50104ac9b279f99e88d848e455262a3fbeb55c50e7	./core/libraries/UnsafeMath.sol
d6cb9a153be4ea9fb2377ef88641ef7979b5cee6933162f1b732d0289e26e1b6	./core/libraries/SwapMath.sol
c2b03bbf6ae73415e9f60fb2bcdad1ee9dbb3ab1f27f9b12384c44d11a5624e0	./core/NoDelegateCall.sol
e24f9173de3fee0e5fa3b5178083b7f079eb964343ca70e54ce805bc9a3b131b	./core/RabbitSwapV3Pool.sol
16a2da5cc7eb69cc3f6a57211677f9d37c2023e47f3b27c4104fc459d8612b00	./core/RabbitSwapV3PoolDeployer.sol
f4d9fac457acead017098c68bf8ee3c66f705aaf8b0f05961f41d491de5402eb	./periphery/libraries/SqrtPriceMathPartial.sol
a1fc6e34c67bac0d3dfe369a758f44dd23e48756b0d63da6f021d16197c30d0d	./periphery/libraries/SafeERC20Namer.sol
ce2c59159a9dd49d0465000ce2f255f79b414497d1c148e0a85970db5096af13	./periphery/libraries/AddressStringUtil.sol
dee558eee89622b59cb62464b6da37ffa8749c3b6132f5e13d19c59713c48665	./periphery/libraries/LiquidityAmounts.sol
5203d004614507d1573345889ee2e763f25b867bcca90aa6b771ddf7063c2d11	./periphery/libraries/HexStrings.sol
42edaa8b6c577bee7a24b2f1d377fa7fb7649526a935040ccdd1a91a7f3b46a0	./periphery/libraries/Path.sol

b811728b2a5081639f7186390533821b9407b71a3172d72fa14ed5c19a15c8fc	./periphery/libraries/PositionKey.sol
7b3426d328990cd02b5e38e18eb1c0d6cba21ef048758891bd57d5af071105b5	./periphery/libraries/PoolAddress.sol
775f5a497f30cbaea40cc00faa0623aba68e185a14f8d0c947a1d7b229907b0c	./periphery/libraries/PositionValue.sol
f6520df5263c8938a53d2a53ee274d959ba63770c6e70c6863a5728a905ad751	./periphery/libraries/ChainId.sol
6ed92f6a129c8eabce14334f135ee44cd015a286c7f353ecf20a26f75d2b7e39	./periphery/libraries/NFTSVG.sol
a576eed4c87828baff63c10bb77c2b25c0d73d2541bbab430fd8027d84590d7f	./periphery/libraries/TokenRatioSortOrder.sol
14897806cf734a0d046d31599c6f33174c29262d9525baa36bd2330ac7cb815b	./periphery/libraries/PoolTicksCounter.sol
9d9b04abb69c4734ab01f37a67205a32c87a50c28de8aaf33f04566e7978dcbb	./periphery/libraries/CallbackValidation.sol
7d02f695d41542209c5aa2b18b4041b53412e494491b3c44a361828261c366fd	./periphery/libraries/TransferHelper.sol
9926456815af16792199f7905b14868553c706b3cc3366333c10b4f65a5b1fbf	./periphery/libraries/OracleLibrary.sol
92ae7ec9071ba606c09954a00a8de4cc2ccf1d8ab24c9670a169a7699b3a7c44	./periphery/libraries/NFTDescriptor.sol
abe5da07d5e9f890fc64ca7b9283fa88a81a0909e4510452bdfb470d4d49bddf	./periphery/libraries/BytesLib.sol
023c9b73c3ec1f4371d03f6c768db2187ab9f3e6e7791c456a718dd538bf8bf2	./periphery/SwapRouter.sol
67e853c5b6fd830703c27c50ac2d61378b733c2208e13713c49e1f3967837387	./periphery/lens/TickLens.sol
f5a21edc2580bf53396761017b02ee1c295eae987eced4955da2ced2f3b8b2a6	./periphery/lens/README.md
22d73470f15c1afda29e934cc1554c4fb9a31854567f924f5cd1c77ef5cc2850	./periphery/lens/QuoterV2.sol

529e895030783bee5a08169006734194b6595fd462cc9a904b5e9adbf183bc0a	./periphery/lens/Quoter.sol
299dfc3cfc4611cde90d5d61eacd119e03d31d1cb3dfd9c7b12839fb527cd901	./periphery/lens/RabbitSwapInterfaceMulticall.sol
d54119eb5481a86f0ca272a8206d766a6f4730bd516e4e36fe6cc7b6ac5e90cd	./periphery/NonfungiblePositionManager.sol
b6296af6dc2269b5c5c9f0ff91588c74708b6797c7412f165bfd063fa5b72d9b	./periphery/NonfungibleTokenPositionDescriptor.sol
48bb499a5e2bb8063788faf42ba0abd71cbd63392aa4d4c12531b530419d6afa	./periphery/base/SelfPermit.sol
a92af8c5c06ded26e1a66af269e10edd6f018aa13cd97ca16e82b1da8dfa0258	./periphery/base/LiquidityManagement.sol
44a19421fcfced88d9af3fee9881e260545b15d12575d2062c88239c87776531	./periphery/base/PoolInitializer.sol
3a69e6dbcf3f5a9a91ac2884373312241d0f5f444fcdb995ad52ff73bf425eaa	./periphery/base/PeripheryImmutableState.sol
c316608e8422486f6eb41624756a729a5cef178ab0db68d21a30b4a261a34bfb	./periphery/base/PeripheryPaymentsWithFee.sol
68cef83e01906a13f4a2bb1c12a9e99fad3e957eea6ddbb54bac30ba3b06a436	./periphery/base/PeripheryPayments.sol
d917dd488471948d666b4c929f9df7a3b4133db6874de2c8c2a1a2e713c0e984	./periphery/base/ERC721Permit.sol
029ad0bcade48ff32da51094a3fb245fd7d8324c4fb4dd20fb4b2614efc9618c	./periphery/base/Multicall.sol
e5ca9a8b6b9e0cafcd9a9966b05228a1572f82fccee396d2e0eff5f8aa9bb1f4	./periphery/base/BlockTimestamp.sol
40877c212ebd04f41a3c582bbbf8ad925f31b2a4f7f129352f55777c8fd584a0	./periphery/base/PeripheryValidation.sol
f4d14cafdc62c73c8f222fd1dc5a11246c58601e681220fe8fb996fb4475e3dd	./vrc25/VRC25Upgradable.sol
63a2df886c6526e2f086b888acd6a4431d809e976b2b41566aa7b87172c3c4f6	./vrc25/VRC25Permit.sol

1740c0422e2328ea00e2e36d7336b5bc06c9d9214a50f16d070edafc06189eae	./vrc25/libraries/SafeMath.sol
a56d67219e7f908e9a9e98b3e1b102bc8ba334c7c7f7ab6b917ead75fc6c6d89	./vrc25/libraries/Address.sol
e11379243981645a7357b43f17bbdd8e50e9bd87d7ffee6493e1865febc07865	./vrc25/libraries/ECDSA.sol
a2e8710f5e9ee59390dac5a8d2ef1511edbeed0d157d5cdf4cb6bff4f1708568	./vrc25/libraries/EIP712.sol
de84b094c0ebfba229dc6b14376a970df8c095a96fdae2038ca0e486e917f377	./vrc25/VRC25.sol
ccd400af4a0296e63163d65738ecfb7a37296d46bab8abed805770cdf69b6049	./vrc25/ERC165.sol

1.3. Audit Methodology

Our security audit process for smart contract includes two steps:

- Smart contract codes are scanned/tested for commonly known and more specific vulnerabilities using public and RK87, our in-house smart contract security analysis tool.
- Manual audit of the codes for security issues. The contracts are manually analyzed to look for any potential problems.

Following is the list of commonly known vulnerabilities that were considered during the audit of the smart contract:

- Integer Overflow and Underflow
- Timestamp Dependence
- Race Conditions
- Transaction-Ordering Dependence
- DoS with (Unexpected) revert
- DoS with Block Gas Limit
- Gas Usage, Gas Limit and Loops
- Redundant fallback function
- Unsafe type Inference
- Reentrancy
- Explicit visibility of functions state variables (external, internal, private and public)
- Logic Flaws

For vulnerabilities, we categorize the findings into categories as listed in table below, depending on their severity level:

SEVERITY LEVEL	DESCRIPTION
CRITICAL	A vulnerability that can disrupt the contract functioning; creates a critical risk to the contract; required to be fixed immediately.
HIGH	A vulnerability that could affect the desired outcome of executing the contract with high impact; needs to be fixed with high priority.
MEDIUM	A vulnerability that could affect the desired outcome of executing the contract with medium impact in a specific scenario; needs to be fixed.
LOW	An issue that does not have a significant impact, can be considered as less important.

Table 1. Severity levels

1.4. Disclaimer

RabbitSwap acknowledges that the security services provided by Verichains, are conducted to the best of their professional abilities but cannot guarantee 100% coverage of all security vulnerabilities. RabbitSwap understands and accepts that despite rigorous auditing, certain vulnerabilities may remain undetected. Therefore, RabbitSwap agrees that Verichains shall not be held responsible or liable, and shall not be charged for any hacking incidents that occur due to security vulnerabilities not identified during the audit process.

1.5. Acceptance Minute

This final report served by Verichains to the RabbitSwap will be considered an Acceptance Minute. Within 7 days, if no any further responses or reports is received from the RabbitSwap, the final report will be considered fully accepted by the RabbitSwap without the signature.

2. AUDIT RESULT

2.1. Overview

The RabbitSwap V3 smart contracts are written in Solidity language and utilize `OpenZeppelin`'s contract libraries, emphasizing modularity and security.

The RabbitSwap V3 smart contracts were written in `Solidity` language, with the required version to be `0.7.6`. They are forked from `Uniswap V3` protocol.

2.1.1. Core Contracts

The core consists of a single factory, a pool deployer, and the many pools the factory will create.

A significant amount of care and attention has been given to gas optimization in the core contracts. The result is a substantial reduction in gas costs for all protocol interactions compared to V2, at the cost of a reduction in code clarity.

2.1.2. Periphery Contracts

The periphery is a constellation of smart contracts designed to support domain-specific interactions with the core. As the Uniswap protocol is a permissionless system, the contracts described below have no special privileges and are only a small subset of possible periphery-like contracts.

2.1.3. VRC25 Contract

`VRC25` is a fork of the official standard for Fungible Tokens in Viction ecosystem. RabbitSwap V3 smart contracts use this contract to support Viction ZeroGas for performing gas-less transaction.

2.1.4. VRC725 Contract

`VRC725` is a fork of the official standard for Non-Fungible Tokens in the Viction ecosystem, which adds the `Enumerable` feature. RabbitSwap V3 smart contracts replace `ERC721Permit` with this contract to support Viction ZeroGas for performing gas-less transaction.

2.2. Findings

During the audit process, the audit team had identified some issues in the source code, along with some recommendations.

RabbitSwap fixed the code, according to Verichains's Draft report.

#	Issue	Severity	Status
2	_ownerOf function could be reverted unexpectedly	LOW	Fixed
3	protocolFeesToken variables have not been updated yet	LOW	Fixed
4	Unused function parameters should be commented out or removed	INFORMATIVE	Fixed

2.2.1. _ownerOf function could be reverted unexpectedly LOW

Affected files:

- vrc725/VRC725.sol

`_ownerOf` was modified from the original one to support enumerable, but it changed the behavior of the function. The original `_ownerOf` function does not revert if the token does not exist. However, the modified `_ownerOf` function reverts if the token does not exist but the dev documents still claims that this function does not revert. This could lead to unexpected behavior in the feature versions or inherited contracts.

```
/**
 * @dev Returns the owner of the `tokenId`. Does NOT revert if token doesn't exist
 */
function _ownerOf(uint256 tokenId) internal view virtual returns (address) {
    return _tokenOwners.get(tokenId, "ERC721: owner query for nonexistent token");
}

// Original _ownerOf function
/**
 * @dev Returns the owner of the `tokenId`. Does NOT revert if token doesn't exist
 */
function _ownerOf(uint256 tokenId) internal view virtual returns (address) {
    return _owners[tokenId];
}
```

RECOMMENDATION

- Update the documentation to reflect the actual behavior of the `_ownerOf` function.
- Consider making the function not revert if the token does not exist to be consistent with the original `_ownerOf` function to avoid unexpected behavior.

UPDATES

- **Nov 15, 2024**: This issue has been acknowledged and fixed by RabbitSwap team.

2.2.2. protocolFeesToken variables have not been updated yet LOW

Affected files:

- core/RabbitSwapV3Pool.sol

`protocolFeesToken0` and `protocolFeesToken1` have not been updated yet in the `swap` function. The `Swap` event is emitted with `protocolFeesToken0` and `protocolFeesToken1` as `uint128` but the actual value is `0`. This could lead to confusion when reading and parsing the event logs.

```
function swap(
    address recipient,
    bool zeroForOne,
    int256 amountSpecified,
    uint160 sqrtPriceLimitX96,
    bytes calldata data
) external override noDelegateCall returns (int256 amount0, int256 amount1) {
    ...
    // to emit in Swap() event
    uint128 protocolFeesToken0 = 0;
    uint128 protocolFeesToken1 = 0;
    ...
    emit Swap(msg.sender, recipient, amount0, amount1, state.sqrtPriceX96, state.liquidity,
state.tick, protocolFeesToken0, protocolFeesToken1);
}
```

RECOMMENDATION

- Update the `protocolFeesToken0` and `protocolFeesToken1` values in the `swap` function to reflect the actual value.
- Consider removing the `protocolFeesToken0` and `protocolFeesToken1` from the `Swap` event if they are not used.

UPDATES

- **Nov 15, 2024**: This issue has been acknowledged and fixed by RabbitSwap team.

2.2.3. Unused function parameters should be commented out or removed INFORMATIVE

Affected files:

- vrc725/libraries/ECDSA.sol
- periphery/SwapRouter.sol

The `errorArg` function parameter in the `_throwError` function is not used; it is recommended to comment it out or remove it to avoid compilation warnings and improve code readability. This approach is better and more appropriate than the current method.

The same goes for the `value` function parameter in the `_estimateFee` function.

```
// vrc725/libraries/ECDSA.sol
function _throwError(RecoverError error, bytes32 errorArg) private pure {
    errorArg;
    require(error == RecoverError.NoError, error == RecoverError.InvalidSignature ?
        "ECDSA: invalid signature" :
        error == RecoverError.InvalidSignatureLength ?
            "ECDSA: invalid signature length" :
            "ECDSA: invalid signature 's' value");
}

// periphery/SwapRouter.sol
function _estimateFee(uint256 value) internal view override returns (uint256) {
    value;
    return minFee();
}
```

RECOMMENDATION

- Update the `protocolFeesToken0` and `protocolFeesToken1` values in the `swap` function to reflect the actual value.
- Consider removing the `protocolFeesToken0` and `protocolFeesToken1` from the `Swap` event if they are not used.

```
// vrc725/libraries/ECDSA.sol
function _throwError(RecoverError error, bytes32 /*errorArg*/) private pure {
    require(error == RecoverError.NoError, error == RecoverError.InvalidSignature ?
        "ECDSA: invalid signature" :
        error == RecoverError.InvalidSignatureLength ?
            "ECDSA: invalid signature length" :
            "ECDSA: invalid signature 's' value");
}

// periphery/SwapRouter.sol
function _estimateFee(uint256 /*value*/) internal view override returns (uint256) {
    return minFee();
}
```

UPDATES

- **Nov 15, 2024**: This issue has been acknowledged and fixed by RabbitSwap team.

3. VERSION HISTORY

Version	Date	Status/Change	Created by
1.0	*Nov 15, 2024*	Public Report	Verichains Lab

Table 2. Report versions history