BEOSIN
Blockchain Security

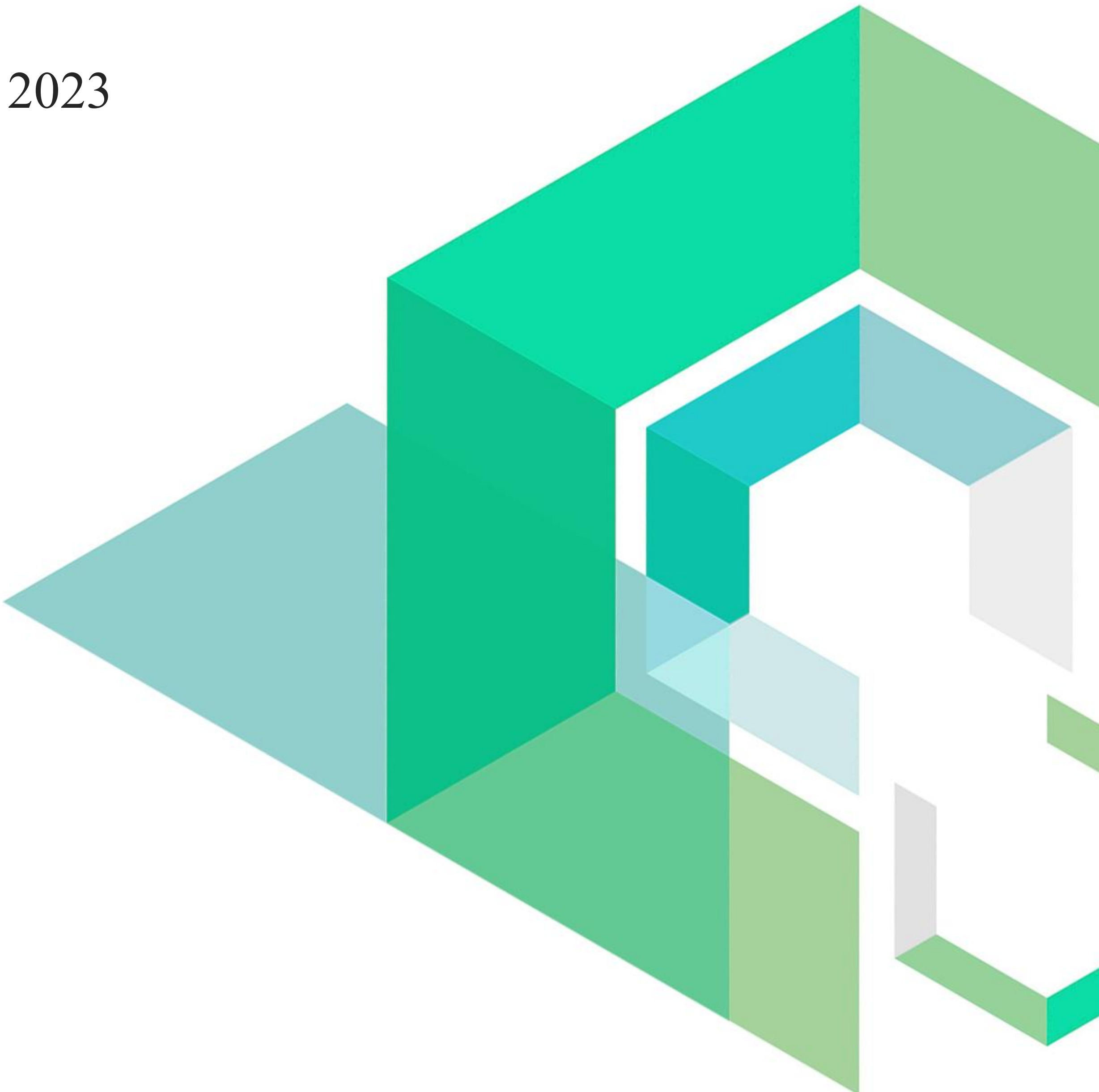
SyncSwap

Smart Contract Security Audit

V1.0

No. 202304231022

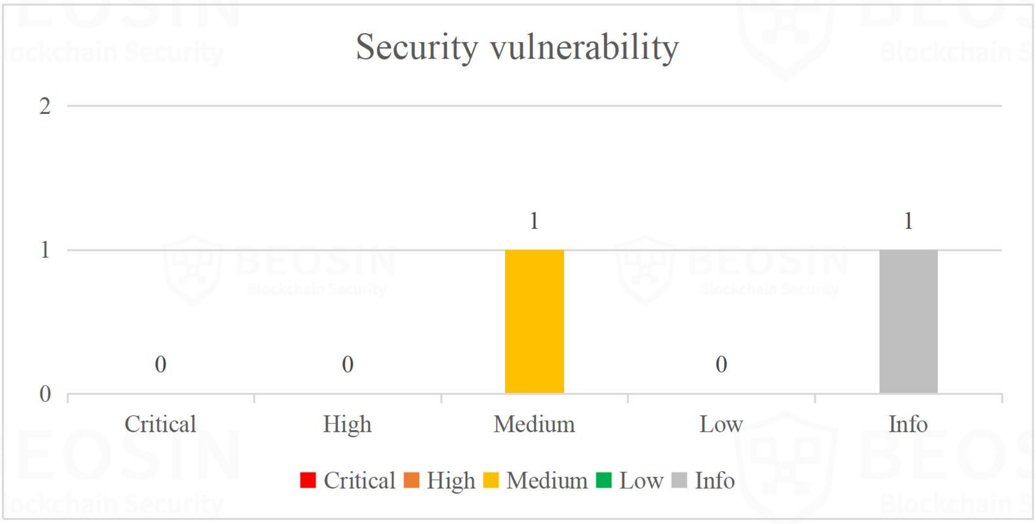
Apr 23th, 2023



Contents

Summary of Audit Results

After auditing, 1 Medium risk and 1 Info items were identified in the SYNCSWAP project. Specific audit details will be presented in the **Findings** section. Users should pay attention to the following aspects when interacting with this project:



***Notes:**

- **Risk Description:**

Users should concern conditions as below:

1. The contract owner can withdraw the assets in the contract by calling *withdrawETH/withdrawERC20*. Not affected by the activation status, the owner can withdraw all assets of the contract at any time. The project party said that in some emergency situations, this function can play a role in rescue. For example, when a user accidentally transfers tokens to a contract, it is necessary to withdraw raised funds, or in the case of urgent replacement of the startup contract or change of parameters, withdrawal of sale tokens, etc. It is recommended that users always pay attention to changes in contract status and other important parameters. Avoid loss of assets.

2. The saleToken and paymentToken address can be changed arbitrarily.

3. The saleToken and paymentToken tokens decimals are different, which may cause problems.

Project Description:

1. Business overview

The SyncSwap smart contract implements the following functions: set the contract deployer as the sole manager of all functions, and allow the owner to set parameters such as whether the refund function is enabled, the allowed hard cap value, the payment token address, and the sales token address. In this contract, the specified tokens are charged to contract contributors as payment tokens, and the contribution funds of each user are recorded, and the contribution is stopped when the hard cap value is reached. At the same time, based on the user's contribution funds, the number of tokens purchased by the user is calculated, and these tokens are distributed to the corresponding contributors after the contribution is over. Finally, the contract also allows owner to withdraw any ERC20 token or ether balance from it.

1 Overview

1.1 Project Overview

Project Name	SyncSwap
Platform	zkSync
Contract address	https://github.com/syncswap/launch-contracts/
Commit Hash	39d01bee6a2e277d7363b407db151bfedc757a93

1.2 Audit Overview

Audit work duration: Apr 21, 2023 –Apr 23, 2023

Audit methods: Formal Verification, Static Analysis, Typical Case Testing and Manual Review.

Audit team: Beosin Security Team.

2 Findings

Index	Risk description	Severity level	Status
SYNCSWAP - 1	Owner permission is too high	Medium	Acknowledged
SYNCSWAP - 2	Check if the precision of saleToken and paymentToken are the same	Info	Acknowledged

Status Notes:

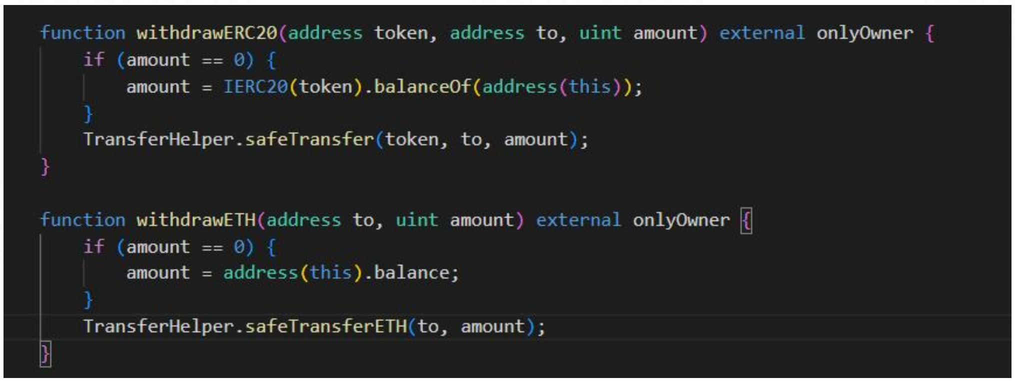
1. SYNCSWAP-1 is not fixed. Project Response this gives us more flexibility when handling the launch since the launch is not meant to be permissionless, and we will work closely with the projects to launch (i.e., the token creator), and the owner permission shouldn't be an issue.

2. SYNCSWAP-2 is not fixed. The project party responded the issue could be scarce in most cases. The launch contract is not meant to be fully permissionless. Therefore every launch on the platform will be manually checked to ensure the token amount and decimals are in the reasonable range to ensure the rounding won 't cause problems.

Finding Details:

[SYNCSWAP - 1] Owner permission is too high

Severity Level	Medium
Type	Business Security
Lines	SyncSwappadToken.sol #L423-L435
Description	The owner can call *withdrawETH/withdrawERC20* to transfer all _paymentToken and saleToken in the contract, then the user will fail to call *withdraw/claim* because there are not enough assets in the contract to send to the user.



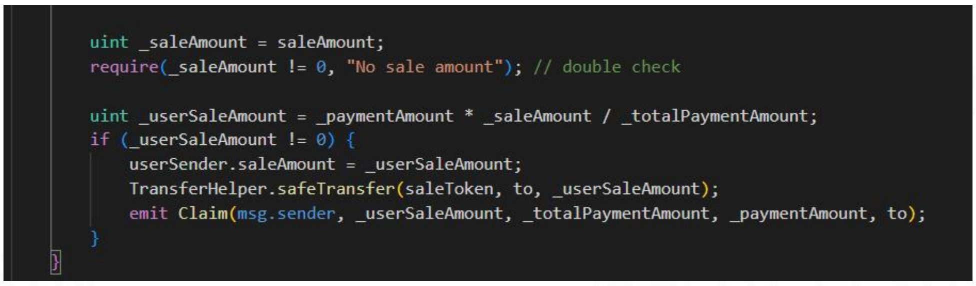
```
function withdrawERC20(address token, address to, uint amount) external onlyOwner {
    if (amount == 0) {
        amount = IERC20(token).balanceOf(address(this));
    }
    TransferHelper.safeTransfer(token, to, amount);
}

function withdrawETH(address to, uint amount) external onlyOwner {
    if (amount == 0) {
        amount = address(this).balance;
    }
    TransferHelper.safeTransferETH(to, amount);
}
```

Figure 1 Source code of *withdrawERC20* and *withdrawETH* function

Recommendations	It is recommended that the owner can only withdraw the assets earned by the contract when calling *withdrawERC20* and *withdrawETH*, such as the paymentToken obtained by the contract after the user claims. Instead of directly transferring all assets of the contract.
Status	Acknowledged.

[SYNCSWAP - 2] Check if the precision of saleToken and paymentToken are the same

Severity Level	Info
Type	Business Security
Lines	SyncSwapLunchPool.sol #L411
Description	Calculate _userSaleAmount = _paymentAmount * _saleAmount / _totalPaymentAmount in the *claim* function. If the precision of saleToken is much greater than that of paymentToken, the resulting _userSaleAmount may be 0.



```
    uint _saleAmount = saleAmount;
    require(_saleAmount != 0, "No sale amount"); // double check

    uint _userSaleAmount = _paymentAmount * _saleAmount / _totalPaymentAmount;
    if (_userSaleAmount != 0) {
        userSender.saleAmount = _userSaleAmount;
        TransferHelper.safeTransfer(saleToken, to, _userSaleAmount);
        emit Claim(msg.sender, _userSaleAmount, _totalPaymentAmount, _paymentAmount, to);
    }
}
```

Figure 2 Source code of *claim* function

Recommendations	It is recommended to check the accuracy when setting the address of saleToken and paymentToken.
Status	Acknowledged.

3 Appendix

3.1 Vulnerability Assessment Metrics and Status in Smart Contracts

3.1.1 Metrics

In order to objectively assess the severity level of vulnerabilities in blockchain systems, this report provides detailed assessment metrics for security vulnerabilities in smart contracts with reference to CVSS 3.1 (Common Vulnerability Scoring System Ver 3.1).

According to the severity level of vulnerability, the vulnerabilities are classified into four levels: "critical", "high", "medium" and "low". It mainly relies on the degree of impact and likelihood of exploitation of the vulnerability, supplemented by other comprehensive factors to determine of the severity level.

Impact / Likelihood	Severe	High	Medium	Low
Probable	Critical	High	Medium	Low
Possible	High	High	Medium	Low
Unlikely	Medium	Medium	Low	Info
Rare	Low	Low	Info	Info

3.1.2 Degree of impact

- **Severe**

Severe impact generally refers to the vulnerability can have a serious impact on the confidentiality, integrity, availability of smart contracts or their economic model, which can cause substantial economic losses to the contract business system, large-scale data disruption, loss of authority management, failure of key functions, loss of credibility, or indirectly affect the operation of other smart contracts associated with it and cause substantial losses, as well as other severe and mostly irreversible harm.

- **High**

High impact generally refers to the vulnerability can have a relatively serious impact on the confidentiality, integrity, availability of the smart contract or its economic model, which can cause a greater economic loss, local functional unavailability, loss of credibility and other impact to the contract business system.

- **Medium**

Medium impact generally refers to the vulnerability can have a relatively minor impact on the confidentiality, integrity, availability of the smart contract or its economic model, which can cause a small amount of economic loss to the contract business system, individual business unavailability and other impact.

- **Low**

Low impact generally refers to the vulnerability can have a minor impact on the smart contract, which can pose certain security threat to the contract business system and needs to be improved.

3.1.4 Likelihood of Exploitation

- **Probable**

Probable likelihood generally means that the cost required to exploit the vulnerability is low, with no special exploitation threshold, and the vulnerability can be triggered consistently.

- **Possible**

Possible likelihood generally means that exploiting such vulnerability requires a certain cost, or there are certain conditions for exploitation, and the vulnerability is not easily and consistently triggered.

- **Unlikely**

Unlikely likelihood generally means that the vulnerability requires a high cost, or the exploitation conditions are very demanding and the vulnerability is highly difficult to trigger.

- **Rare**

Rare likelihood generally means that the vulnerability requires an extremely high cost or the conditions for exploitation are extremely difficult to achieve.

3.1.5 Fix Results Status

Status	**Description**
Fixed	The project party fully fixes a vulnerability.
Partially Fixed	The project party did not fully fix the issue, but only mitigated the issue.
Acknowledged	The project party confirms and chooses to ignore the issue.

3.2 Audit Categories

No.	Categories	Subitems
1	Coding Conventions	Compiler Version Security
		Deprecated Items
		Redundant Code
		require/assert Usage
		Gas Consumption
2	General Vulnerability	Integer Overflow/Underflow
		Reentrancy
		Pseudo-random Number Generator (PRNG)
		Transaction-Ordering Dependence
		DoS (Denial of Service)
		Function Call Permissions
		call/delegatecall Security
		Returned Value Security
		tx.origin Usage
		Replay Attack
		Overriding Variables
		Third-party Protocol Interface Consistency
3	Business Security	Business Logics
		Business Implementations
		Manipulable Token Price
		Centralized Asset Control
		Asset Tradability
		SyncSwaptrage Attack

Beosin classified the security issues of smart contracts into three categories: Coding Conventions, General Vulnerability, Business Security. Their specific definitions are as follows:

● **Coding Conventions**

Audit whether smart contracts follow recommended language security coding practices. For example, smart contracts developed in Solidity language should fix the compiler version and do not use deprecated keywords.

- **General Vulnerability**

General Vulnerability include some common vulnerabilities that may appear in smart contract projects. These vulnerabilities are mainly related to the characteristics of the smart contract itself, such as integer overflow/underflow and denial of service attacks.

- **Business Security**

Business security is mainly related to some issues related to the business realized by each project, and has a relatively strong pertinence. For example, whether the lock-up plan in the code match the white paper, or the flash loan attack caused by the incorrect setting of the price acquisition oracle.

*Note that the project may suffer stake losses due to the integrated third-party protocol. This is not something Beosin can control. Business security requires the participation of the project party. The project party and users need to stay vigilant at all times.

3.3 Disclaimer

The Audit Report issued by Beosin is related to the services agreed in the relevant service agreement. The Project Party or the Served Party (hereinafter referred to as the "Served Party") can only be used within the conditions and scope agreed in the service agreement. Other third parties shall not transmit, disclose, quote, rely on or tamper with the Audit Report issued for any purpose.

The Audit Report issued by Beosin is made solely for the code, and any description, expression or wording contained therein shall not be interpreted as affirmation or confirmation of the project, nor shall any warranty or guarantee be given as to the absolute flawlessness of the code analyzed, the code team, the business model or legal compliance.

The Audit Report issued by Beosin is only based on the code provided by the Served Party and the technology currently available to Beosin. However, due to the technical limitations of any organization, and in the event that the code provided by the Served Party is missing information, tampered with, deleted, hidden or subsequently altered, the audit report may still fail to fully enumerate all the risks.

The Audit Report issued by Beosin in no way provides investment advice on any project, nor should it be utilized as investment suggestions of any type. This report represents an extensive evaluation process designed to help our customers improve code quality while mitigating the high risks in blockchain.

3.4 About Beosin

Beosin is the first institution in the world specializing in the construction of blockchain security ecosystem. The core team members are all professors, postdocs, PhDs, and Internet elites from world-renowned academic institutions. Beosin has more than 20 years of research in formal verification technology, trusted computing, mobile security and kernel security, with overseas experience in studying and collaborating in project research at well-known universities. Through the security audit and defense deployment of more than 2,000 smart contracts, over 50 public blockchains and wallets, and nearly 100 exchanges worldwide, Beosin has accumulated rich experience in security attack and defense of the blockchain field, and has developed several security products specifically for blockchain.