SALUS SECURITY

MAR 2025

CODE SECURITY ASSESSMENT

FOUR

Overview

Project Summary

- Name: Four - smart cannon router
- Platform: EVM-compatible chains
- Language: Solidity, Rust
- Audit Range: See Appendix - 1

Project Dashboard

Application Summary

Name	Four - smart cannon router
Version	v3
Type	Solidity
Dates	Mar 12 2025
Logs	Feb 12 2025; Feb 13 2025; Mar 12 2025

Vulnerability Summary

Total High-Severity issues	0
Total Medium-Severity issues	2
Total Low-Severity issues	3
Total informational issues	4
Total	9

Contact

E-mail: support@salusec.io

Risk Level Description

High Risk	The issue puts a large number of users' sensitive information at risk, or is reasonably likely to lead to catastrophic impact for clients' reputations or serious financial implications for clients and users.
Medium Risk	The issue puts a subset of users' sensitive information at risk, would be detrimental to the client's reputation if exploited, or is reasonably likely to lead to a moderate financial impact.
Low Risk	The risk is relatively small and could not be exploited on a recurring basis, or is a risk that the client has indicated is low impact in view of the client's business circumstances.
Informational	The issue does not pose an immediate risk, but is relevant to security best practices or defense in depth.

Content

Introduction

1.1 About SALUS

At Salus Security, we are in the business of trust.

We are dedicated to tackling the toughest security challenges facing the industry today. By building foundational trust in technology and infrastructure through security, we help clients to lead their respective industries and unlock their full Web3 potential.

Our team of security experts employ industry-leading proof-of-concept (PoC) methodology for demonstrating smart contract vulnerabilities, coupled with advanced red teaming capabilities and a stereoscopic vulnerability detection service, to deliver comprehensive security assessments that allow clients to stay ahead of the curve.

In addition to smart contract audits and red teaming, our Rapid Detection Service for smart contracts aims to make security accessible to all. This high calibre, yet cost-efficient, security tool has been designed to support a wide range of business needs including investment due diligence, security and code quality assessments, and code optimisation.

We are reachable on Telegram (https://t.me/salusec), Twitter (https://twitter.com/salus_sec), or Email (support@salusec.io).

1.2 Audit Breakdown

The objective was to evaluate the repository for security-related issues, code quality, and adherence to specifications and best practices. Possible issues we looked for included (but are not limited to):

- Risky external calls
- Integer overflow/underflow
- Transaction-ordering dependence
- Timestamp dependence
- Access control
- Call stack limits and mishandled exceptions
- Number rounding errors
- Centralization of power
- Logical oversights and denial of service
- Business logic specification
- Code clones, functionality duplication

1.3 Disclaimer

Note that this security audit is not designed to replace functional tests required before any software release and does not give any warranties on finding all possible security issues with the given smart contract(s) or blockchain software, i.e., the evaluation result does not guarantee the nonexistence of any further findings of security issues.

Findings

2.1 Summary of Findings

ID	Title	Severity	Category	Status
1	Incorrect token transfer processing	Medium	Business Logic	Resolved
2	Incorrect swapAmount calculation	Medium	Business Logic	Resolved
3	Fee authority address manipulation	Low	Access Control	Acknowledged
4	Centralization risk	Low	Business Logic	Acknowledged
5	May be divided by zero when calculating threshold	Low	Numerics	Resolved
6	Use call instead of transfer for native tokens transfer	Informational	Configuration	Acknowledged
7	Misspelled function name	Informational	Business logic	Acknowledged
8	Missing two-step transfer ownership pattern	Informational	Inconsistency	Acknowledged
9	Use of floating pragma	Informational	Configuration	Acknowledged

2.2 Notable Findings

Significant flaws that impact system confidentiality, integrity, or availability are listed below.

1. Incorrect token transfer processing

Severity: Medium	Category: Business Logic

Target:
- base/smart-cannon-router-v2.0-base.sol
- eth & bsc/smart-cannon-router-v2.0.sol

Description

base/smart-cannon-router-v2.0-base.sol:L88 - L104

```
if(isWeth(token0)) {
    if (amountIn > 0){
        require(msg.value >= amountIn, "msg.value Insufficient");
        // 买入税收取
        ethFee = SafeUtils.sub(msg.value, amountIn);
        swapAmount = amountIn;
    } else {
        require(msg.value >= amountInMax, "msg.value Insufficient");
        // 买入税收取
        ethFee = SafeUtils.sub(msg.value, amountInMax);
        swapAmount = amountInMax;
    }
    IWETH(token0).deposit{value: msg.value}();
} else {
    SafeUtils.safeTransferFrom(token0, tx.origin, address(this), amountIn);
    amountIn = IERC20(token0).balanceOf(address(this)) - token0Balance;
}
```

The `` `execute()` `` function calls `` `safeTransferFrom()` `` to transfer tokens to the contract when `` `token0` `` is not WETH, but there is a problem here. When `` `token0` `` is not `WETH` and the user specifies `` `amountInMax` ``, `` `amountIn` `` is 0, which is equal to not transferring the tokens to the contract, and when `` `swap` `` is performed, there will be an error that the balance of tokens is not enough.

Recommendation

It is recommended that transfer operations be adjusted according to business logic.

Status

This issue has been resolved by the team.

2. Incorrect swapAmount calculation	
Severity: Medium	Category: Business Logic
Target: - base/smart-cannon-router-v2.0-base.sol - eth & bsc/smart-cannon-router-v2.0.sol	

Description

When `WETH` is used as the underlying asset for buying and selling, `swapAmount` records the inflow or outflow of `WETH`. When a user sells, a selling tax is collected. When `feeRate` is 0, the code will not execute the corresponding if branch, thus `swapAmount` will not be correctly recorded.

base/smart-cannon-router-v2.0-base.sol:L175 - L193

```
if(token1BalanceNew > token1Balance) {
    uint256 tokenReturn = token1BalanceNew - token1Balance;
    if(isWeth(token1)) {
        IWETH(token1).withdraw(tokenReturn);

        // 卖出税收取
        if(feeRate > 0) {
            if(feeRate >= 1000) {
                ethFee = tokenReturn;
            } else {
                ethFee = SafeUtils.div(SafeUtils.mul(tokenReturn, feeRate),1000);
            }
            swapAmount = swapAmount + tokenReturn;
            feeType = 1;
        }
    } else {
        SafeUtils.safeTransfer(token1, tx.origin, tokenReturn);
    }
}
```

This will not substantially impact the contract's execution, but it will trigger an event. If backend scripts rely on this event, it may lead to incorrect result recording.

Recommendation

Consider correctly calculating `swapAmount` in the scenario where `feeRate` is 0.

Status

This issue has been resolved by the team.

3. Fee authority address manipulation	
Severity: Low	Category: Access Control
Target: - solana/lib.rs	

Description

The `initialize_fee_accounts()` function allows anyone to call it and set the `fee_authority` to any address. Once set to an incorrect address or maliciously initialized by a bad actor, all fees generated subsequently by swaps can only be withdrawn by that address.
solana/lib.rs:L132 - L145

```
pub fn initialize_fee_accounts(
    ctx: Context<InitializeFeeAccounts>,
    fee_authority: Pubkey,
) -> Result<()> {
    // Only initialize fee account once
    let fee_account: &mut Account<FeeAccount> = &mut ctx.accounts.fee_account;
    if fee_account.is_initialized {
        return Err(MonicaError::NoWay.into());
    }
    // Set fee authority during initialization
    fee_account.fee_authority = fee_authority;
    fee_account.is_initialized = true;
    Ok(())
}
```

Recommendation

Consider adding permission restrictions to this function, allowing only specific addresses to call it.

Status

This issue has been acknowledged by the team.

4. Centralization risk	
Severity: Low	Category: Centralization
Target: - base/smart-cannon-router-v2.0-base.sol - eth & bsc/smart-cannon-router-v2.0.sol	

Description

The contract contains the address of the authority `owner`. The `owner` has the right to withdraw all funds in the contract including fees. If the private key of `owner` is compromised, the attacker can withdraw all the funds in the contract.

Recommendation

We recommend transferring privileged accounts to multi-sig accounts with timelock governors for enhanced security. This ensures that no single person has full control over the accounts and that any changes must be authorized by multiple parties.

Status

This issue has been acknowledged by the team.

5. May be divided by zero when calculating threshold	
Severity: Low	Category: Numerics
Target: - solana/lib.rs	

Description

The second if statement allows `fee_percentage` to be 0, but before this, when calculating the `threshold`, a division by zero situation will occur.
solana/lib.rs:L34 - L42

```
// Calculate the threshold to avoid amount_in is too small to deduct fee
let threshold = 10000_u64.checked_div(fee_percentage).ok_or_else(|| {
    MonicaError::MathOverflow
})?;
// If fee_percentage is 0 or amount_in is too small, swap directly
if fee_percentage == 0 || threshold > amount_in {
    instructions::swap_in(ctx, amount_in, minimum_amount_out)?;
    return Ok(());
}
```

Recommendation

Consider allowing `threshold` calculation only when `fee_percentage` is not zero.

Status

This issue has been resolved by the team.

2.3 Informational Findings

6. Use call instead of transfer for native tokens transfer	
Severity: Informational	Category: Business logic
Target: - base/smart-cannon-router-v2.0-base.sol - eth & bsc/smart-cannon-router-v2.0.sol	

Description

The `transfer` function is not recommended for sending native tokens due to its 2300 gas unit limit which may not work with smart contract wallets or multi-sig. Instead, `call` can be used to circumvent the gas limit.

Recommendation

Consider using a `call` instead of `transfer` for sending native tokens.

Status

This issue has been acknowledged by the team.

7. Misspelled function name	
Severity: Informational	Category: Inconsistency
Target: - base/smart-cannon-router-v2.0-base.sol - eth & bsc/smart-cannon-router-v2.0.sol	

Description

The name of the function `isSupporedDexRouter()` is misspelled, where `Suppored` should be `Supported`.

The name of the contract `SmartConnonSwapProxy` and `SmartConnonBaseSwapProxy` is misspelled, where `connon` should be `cannon`.

Recommendation

Consider correcting the spelling error.

Status

This issue has been acknowledged by the team.

8. Missing two-step transfer ownership pattern	
Severity: Informational	Category: Business logic
Target: - base/smart-cannon-router-v2.0-base.sol - eth & bsc/smart-cannon-router-v2.0.sol	

Description

The `` `SmartConnonSwapProxy` `` and `` `SmartConnonBaseSwapProxy` `` contracts inherit from the `` `OwnableUpgradeable` `` contract. This contract does not implement a two-step process for transferring ownership. Thus, ownership of the contract can easily be lost when making a mistake in transferring ownership.

Recommendation

Consider using the Ownable2StepUpgradeable contract from OpenZeppelin instead.

Status

This issue has been acknowledged by the team.

9. Use of floating pragma	
Severity: Informational	Category: Configuration
Target: - All	

Description

```
pragma solidity ^0.8.0;
pragma solidity ^0.8.10;
```

All contracts use floating compiler versions `` `^0.8.0` `` and `` `^0.8.10` ``.

Using a floating pragma `` `^0.8.0` `` and `` `^0.8.10` `` statements is discouraged, as code may compile to different bytecodes with different compiler versions. Use a locked pragma statement to get a deterministic bytecode. Also use the latest Solidity version to get all the compiler features, bug fixes and optimizations.

Recommendation

It is recommended to use a locked Solidity version throughout the project. It is also recommended to use the most stable and up-to-date version.

Status

This issue has been acknowledged by the team.

Appendix

Appendix 1 - Files in Scope

This audit covered the following files:

File	SHA-1 hash
contracts/base/Commands.sol	9c0bd5be30f1e449c2bee7ddb48a00b9e97721d6
contracts/base/smart-cannon-router-v2.0-base.sol	d682ab2623da33b586bfd46fcf9e209845b61f56
contracts/eth & bsc/Commands.sol	9c0bd5be30f1e449c2bee7ddb48a00b9e97721d6
contracts/eth & bsc/smart-cannon-router-v2.0.sol	07986026ca874c96f121dc5cae240f03f460b049
contracts/solana/lib.rs	d45a8fa9ffea9522e27e41629cf90da03875f02f

And we reviewed the following fixed files:

File	SHA-1 hash
contracts/base/Commands.sol	9c0bd5be30f1e449c2bee7ddb48a00b9e97721d6
contracts/base/smart-cannon-router-v2.0-base.sol	47ce69419921a76dd39d6718a9f5b8732eba0f57
contracts/bsc/Commands.sol	9c0bd5be30f1e449c2bee7ddb48a00b9e97721d6
contracts/bsc/smart-cannon-router-v2.0-bsc.sol	58f7af93ad937429178ab989bc8901a465a3321d
contracts/eth/Commands.sol	d45a8fa9ffea9522e27e41629cf90da03875f02f
contracts/eth/smart-cannon-router-v2.0-eth.sol	e226b413ee794055148a01ab4bdbd73867709810
contracts/sol/monica_v0.2.0/src/lib.rs	f5109e899ad473d835dfc6e979e1c4c71f149b15