# Introduction

A smart contract security audit review of the init captial protocol was done by sparkware's auditor, 0xladboy233, with a focus on the security aspects of the application's implementation.

# About Sparkware security

Sparkware security offers cutting edge and affordable smart contract auditing solution. The auditors get reputatoin and skill by consistently get top place in bug bounty and audit competition.
Our past audit prject including optimism and notional finance and graph protocol. Below is a list of comprehensive security review and research: https://github.com/JeffCX/Sparkware-audit-portfolio

# Disclaimer

A smart contract security review can never verify the complete absence of vulnerabilities. This is a time, resource and expertise bound effort where I try to find as many vulnerabilities as possible. We can not guarantee 100% security after the review or even if the review will find any problems with your smart contracts. Subsequent security reviews, bug bounty programs and on-chain monitoring are strongly recommended.

# About **Protocol**

Init Captial is lending protocol in mantle network. the scope of audit focus on code4rena audit fix review.

# Scope

https://github.com/init-capital/init-private-audit/tree/fix-c4

# Commit hash

a95692b5a2d4393a285cf6510005f2087a072563

# R-01: Remove reward tokens does not emit events

it is recommended to emit a events when removing the reward token in WLPMoeMasterChef.sol

# R-02: should document the behavior of updateOrder

While the updateOrder logic is refractored into a two step process: cancel the active order and then create a new order to resolve the attack vector of order creator frontrun order filler,

It is highly recommend that this behavior should be documented clearly because user may not anticipate updateOrder create a new order.

It is also recommended to rename the function to "cancelAndCreateOrder" to better reflect the internal logic for external integration and dapp users.

# Fix Review and mitigation

### [Resolved] - MarginTradingHook#updateOrder lacks acccess control

updateOrder code is refractored to cancel th order and and then create a new order.

the access control is in function canceOrder, initPosition id is validated

```
    _require(order.initPosId == initPosId, Errors.INVALID_INPUT);
```

so this issue is resolved.

## [Resolved] – token can be updated to arbitrary address

updateOrder code is refractored to cancel the and and then create a new order.

when creating a new order, token address is validated

```
_require(_tokenOut == marginPos.baseAsset || _tokenOut ==
marginPos.quoteAsset, Errors.INVALID_INPUT);
```

so this issue is resolved.

## [Resolved] – fillOrder executor can be front-run by the order creator by changing order's limitPrice_e36

updateOrder code is refractored to cancel the and and then create a new order.

so the issue is resolved, if the order creator wants to frontrun the order filler, the order id will not match.

## [Acknowledged] – MarginTradingHook users could potentially be DOSed

the sponsor will highlight it in the doc to emphasis that the marginTradingHook caller should be able to receive ETH and handle the refund token gracefully, no matter the caller is a smart contract address or EOA account.

## [Resolved] – Fill order cannot cancel order properly

the issue is mitigated by updating the order id status directly when the collateral amount is empty

```
        __orders[_orderId].status = OrderStatus.Cancelled;
```

## [Resolved] – SwapType.CloseExactOut balance check too strict can be DOSed

the issue is mitgated by changing the amount check from == to >=

```
    if (swapInfo.swapType == SwapType.CloseExactOut) {
        // slippage control to make sure that swap helper swap correctly
        _require(swapAmtOut >= swapInfo.amtOut, Errors.SLIPPAGE_CONTROL);
        amtSwapped -= IERC20(swapInfo.tokenIn).balanceOf(address(this));
    }
```

## [Acknowledged] – lack of emergency withdraw in WLPMoeWrapper

Sponsor acknowledged the finding:

For handling emergency withdraw or poison tokens in WLpMasterChef, this will not be implemented in the contract level. We will monitor such scenario off-chain.

In the current codebase, admin can remove reward token, which does mitigate the poison reward token and reward token out of gas issue.

## [Resolved] – LsdApi3OracleReader.sol oracle does not handle future timestamp

the issue is mitigated by make sure only check the time freshenss if the timestamp comes from past, otherwise we can assume the price oracle is fresh.