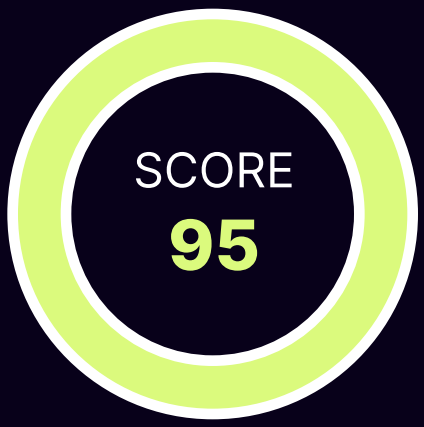
SMART CONTRACT AUDIT

August 10th 2023 | v. 1.0

Security Audit Score

PASS

Zokyo Security has concluded that this smart contract passes security qualifications to be listed on digital asset exchanges.



TECHNICAL SUMMARY

This document outlines the overall security of the Ondo Finance smart contracts evaluated by the Zokyo Security team.

The scope of this audit was to analyze and document the Ondo Finance smart contracts codebase for quality, security, and correctness.

Contract Status

There were 0 critical issues found during the audit. (See Complete Analysis)

It should be noted that this audit is not an endorsement of the reliability or effectiveness of the contracts but rather limited to an assessment of the logic and implementation. In order to ensure a secure contract that can withstand the Ethereum network's fast-paced and rapidly changing environment, we recommend that the Ondo Finance team put in place a bug bounty program to encourage further active analysis of the smart contracts.

Table of Contents

AUDITING STRATEGY AND TECHNIQUES APPLIED

The source code of the smart contract was taken from the Ondo Finance repository:
https://github.com/ondoprotocol/rwa-internal

Full repo EXCLUDING:

- https://github.com/ondoprotocol/rwa-internal/tree/main/contracts/external
- https://github.com/ondoprotocol/rwa-internal/tree/main/contracts/kyc
- https://github.com/ondoprotocol/rwa-internal/tree/main/contracts/rwaOracles
- https://github.com/ondoprotocol/rwa-internal/blob/main/contracts/Proxy.sol
- https://github.com/ondoprotocol/rwa-internal/tree/main/lido
- https://github.com/ondoprotocol/rwa-internal/blob/main/contracts/usdy/allowlist/AllowlistProxy.sol

Commit - f855602e2c047e1658a2c0b5a2f0c87e2ef756e5
The last commit -a598f0532e65c6319e2d1a3d228f106ed25ec9e3

Within the scope of this audit, the team of auditors reviewed the following contract(s):

- InstantMintTimeBasedRateLimiter.sol
- RWAHubInstantMints.sol
- Pricer.sol
- RWAHubOffChainRedemptions.sol
- usdy
- RWAHub.sol
- ommf
- ommfManager.sol ommf_token
- wrappedOMMF
- ./ommf/ommf_token:
- ommf.sol
- ommf_factory.sol
- ./ommf/wrappedOMMF:
- wOMMF.sol
- wOMMF_factory.sol
- ISanctionsListClient.sol
- SanctionsListClient.sol
- SanctionsListClientUpgradeable.sol
- USDY.sol
- USDYFactory.sol
- USDYManager.sol allowlist
- blocklist./usdy/allowlist:
- AllowlistClient.sol
- AllowlistFactory.sol
- AllowlistClientUpgradeable.sol
- AllowlistUpgradeable.sol
- Blocklist.sol
- BlocklistClient.sol
- BlocklistClientUpgradeable.sol
- contracts/ommf/ommf_token/ommf_rebaseSetter.sol
- contracts/ousg/ousgManager.sol
- contracts/RWAHubNonStableInstantMints.sol
- contracts/interfaces/IRWAHubNonStableInstantMint.sol

During the audit, Zokyo Security ensured that the contract:

- Implements and adheres to the existing standards appropriately and effectively;
- The documentation and code comments match the logic and behavior;
- Distributes tokens in a manner that matches calculations;
- Follows best practices, efficiently using resources without unnecessary waste;
- Uses methods safe from reentrance attacks;
- Is not affected by the most resent vulnerabilities;
- Meets best practices in code readability, etc.

Zokyo's Security Team has followed best practices and industry-standard techniques to verify the implementation of Ondo Finance smart contracts. To do so, the code is reviewed line-by-line by our smart contract developers, documenting any issues as they are discovered. In summary, our strategies consist largely of manual collaboration between multiple team members at each stage of the review:

01	Due diligence in assessing the overall code quality of the codebase.
02	Cross-comparison with other, similar smart contracts by industry leaders.
03	Thorough manual review of the codebase line by line.

Executive Summary

Throughout the audit, the Zokyo team identified various issues, including high, medium, low, and informational concerns, all of which are extensively detailed in the "Complete Analysis" section. However, no critical issues were uncovered.

STRUCTURE AND ORGANIZATION OF THE DOCUMENT

For the ease of navigation, the following sections are arranged from the most to the least critical ones. Issues are tagged as “Resolved” or “Unresolved” or “Acknowledged” depending on whether they have been fixed or addressed. Acknowledged means that the issue was sent to the Ondo Finance team and the Ondo Finance team is aware of it, but they have chosen to not solved it. The issues that are tagged as “Verified” contain unclear or suspicious functionality that either needs explanation from the Client or remains disregarded by the Client. Furthermore, the severity of each issue is written as assessed by the risk of exploitation or other unexpected or otherwise unsafe behavior:

Critical

The issue affects the contract in such a way that funds may be lost, allocated incorrectly, or otherwise result in a significant loss.

High

The issue affects the ability of the contract to compile or operate in a significant way.

Medium

The issue affects the ability of the contract to operate in a way that doesn’t significantly hinder its behavior.

Low

The issue has minimal impact on the contract’s ability to operate.

Informational

The issue has no impact on the contract’s ability to operate.

COMPLETE ANALYSIS

FINDINGS SUMMARY

#	Title	Risk	Status
1	Anyone can claim excess	High	Resolved
2	The incorrect variable used to compare	Medium	Resolved
3	Initial mints are paused	Low	Acknowledged
4	Claim excess is paused	Low	Acknowledged
5	Instant redemption is paused	Low	Acknowledged
6	SPDX license not specified	Informational	Resolved
7	Unused function parameter	Informational	Resolved
8	`wommf` should be immutable	Informational	Resolved
9	`rwaOracle` should be immutable	Informational	Resolved
10	`collateral` should be immutable	Informational	Resolved
11	`decimalsMultiplier` should be immutable	Informational	Resolved

HIGH | RESOLVED

Anyone can claim excess

In the RWAHubNonStableInstantMints contract there is no check for the msg.sender to be authorized to claim the excess of the deposit. The only check there is to be KYC'd. Therefore, anyone who went through the KYC could get claim excess

Recommendation:
add a check for the msg.sender to be authorized by the depositor.user to claim the excess or at least mint to the depositor user, not to msg.sender

MEDIUM | RESOLVED

The incorrect variable used to compare

In the contract RWAHubOffChainRedemptions the function requestRedemptionServicedOffchain is using the minimumRedemptionAmount to compare the amountRWATokenToRedeem while the minimumOffChainRedemptionAmount is unused

Recommendation:
make sure minimumOffChainRedemptionAmount is needed.

LOW | ACKNOWLEDGED

Initial mints are paused

At the contract deployment, the initial mints of the RWAHubNonStableInstantMints and RWAHubInstantMints are paused by default which may lead to excess calls to unpause the function before it actually starts working

Recommendation:
initialize the instantMintPaused with false if mints need to be ready right after deployment

LOW | ACKNOWLEDGED

Claim excess is paused

At the contract deployment, the claimExcess function of the RWAHubNonStableInstantMints is paused by default, which may lead to an excess call to unpause function before it actually starts working

Recommendation:
initialize the claimExcessPaused with false if claims need to be ready right after deployment

LOW | ACKNOWLEDGED

instant redemption is paused

At the contract deployment, the claimExcess function of the RWAHubInstantMints is paused by default, which may lead to an excess call to unpause function before it actually starts working

Recommendation:
initialize the instantRedemptionPaused with false if claims need to be ready right after deployment

INFORMATIONAL | RESOLVED

SPDX license not specified

The ommf.sol contract doesn't contain the SPDX license header

Recommendation:
add the SPDX license to ommf.sol

INFORMATIONAL | RESOLVED

Unused function parameter

In the contract ommf.sol on the line 588 there is declared a variable which is not used in the code.

Recommendation:
use "_" instead of the named argument

INFORMATIONAL | RESOLVED

`wommf` should be immutable

In the contract ommfManager.sol on the line 25 there is a state variable which value is assigned during in the constructor and not defined elsewhere in the code.

Recommendation:
add the `immutable` attribute to state variables that never change or are set only in the constructor

INFORMATIONAL | RESOLVED

`rwaOracle` should be immutable

In the contract Pricer.sol on the line 45 there is a state variable which value is assigned during in the constructor and not defined elsewhere in the code.

Recommendation:
add the `immutable` attribute to state variables that never change or are set only in the constructor

INFORMATIONAL | RESOLVED

`collateral` should be immutable

In the contract RWAHub.sol on the line 25 there is a state variable which value is assigned during in the constructor and not defined elsewhere in the code.

Recommendation:
add the `immutable` attribute to state variables that never change or are set only in the constructor

INFORMATIONAL | RESOLVED

`decimalsMultiplier` should be immutable

In the contract RWAHub.sol on the line 66 there is a state variable which value is assigned during in the constructor and not defined elsewhere in the code.

Recommendation:
add the `immutable` attribute to state variables that never change or are set only in the constructor

	• **InstantMintTimeBasedRateLimiter.sol** • **RWAHubInstantMints.sol** • **Pricer.sol** • **RWAHubOffChainRedemptions.sol** • **usdy** • **RWAHub.sol** • **ommf**
Re-entrancy	Pass
Access Management Hierarchy	Pass
Arithmetic Over/Under Flows	Pass
Unexpected Ether	Pass
Delegatecall	Pass
Default Public Visibility	Pass
Hidden Malicious Code	Pass
Entropy Illusion (Lack of Randomness)	Pass
External Contract Referencing	Pass
Short Address/ Parameter Attack	Pass
Unchecked CALL Return Values	Pass
Race Conditions / Front Running	Pass
General Denial Of Service (DOS)	Pass
Uninitialized Storage Pointers	Pass
Floating Points and Precision	Pass
Tx.Origin Authentication	Pass
Signatures Replay	Pass
Pool Asset Security (backdoors in the underlying ERC-20)	Pass

	• **ommfManager.sol ommf_token** • **wrappedOMMF** • **./ommf/ommf_token:** • **ommf.sol** • **ommf_factory.sol** • **./ommf/wrappedOMMF:** • **wOMMF.sol**
Re-entrancy	Pass
Access Management Hierarchy	Pass
Arithmetic Over/Under Flows	Pass
Unexpected Ether	Pass
Delegatecall	Pass
Default Public Visibility	Pass
Hidden Malicious Code	Pass
Entropy Illusion (Lack of Randomness)	Pass
External Contract Referencing	Pass
Short Address/ Parameter Attack	Pass
Unchecked CALL Return Values	Pass
Race Conditions / Front Running	Pass
General Denial Of Service (DOS)	Pass
Uninitialized Storage Pointers	Pass
Floating Points and Precision	Pass
Tx.Origin Authentication	Pass
Signatures Replay	Pass
Pool Asset Security (backdoors in the underlying ERC-20)	Pass

	• **wOMMF_factory.sol** • **ISanctionsListClient.sol** • **SanctionsListClient.sol** • **SanctionsListClientUpgradeable.sol** • **USDY.sol** • **USDYFactory.sol** • **USDYManager.sol allowlist**
Re-entrancy	Pass
Access Management Hierarchy	Pass
Arithmetic Over/Under Flows	Pass
Unexpected Ether	Pass
Delegatecall	Pass
Default Public Visibility	Pass
Hidden Malicious Code	Pass
Entropy Illusion (Lack of Randomness)	Pass
External Contract Referencing	Pass
Short Address/ Parameter Attack	Pass
Unchecked CALL Return Values	Pass
Race Conditions / Front Running	Pass
General Denial Of Service (DOS)	Pass
Uninitialized Storage Pointers	Pass
Floating Points and Precision	Pass
Tx.Origin Authentication	Pass
Signatures Replay	Pass
Pool Asset Security (backdoors in the underlying ERC-20)	Pass

	• **blocklist./usdy/allowlist:** • **AllowlistClient.sol** • **AllowlistFactory.sol** • **AllowlistClientUpgradeable.sol** • **AllowlistUpgradeable.sol** • **Blocklist.sol** • **BlocklistClient.sol**
Re-entrancy	Pass
Access Management Hierarchy	Pass
Arithmetic Over/Under Flows	Pass
Unexpected Ether	Pass
Delegatecall	Pass
Default Public Visibility	Pass
Hidden Malicious Code	Pass
Entropy Illusion (Lack of Randomness)	Pass
External Contract Referencing	Pass
Short Address/ Parameter Attack	Pass
Unchecked CALL Return Values	Pass
Race Conditions / Front Running	Pass
General Denial Of Service (DOS)	Pass
Uninitialized Storage Pointers	Pass
Floating Points and Precision	Pass
Tx.Origin Authentication	Pass
Signatures Replay	Pass
Pool Asset Security (backdoors in the underlying ERC-20)	Pass

- **BlocklistClientUpgradeable.sol**
- **contracts/ommf/ommf_token/ommf_rebaseSetter.sol**
- **contracts/ousg/ousgManager.sol**
- **contracts/RWAHubNonStableInstantMints.sol**
- **contracts/interfaces/IRWAHubNonStableInstantMint.sol**

Re-entrancy	Pass
Access Management Hierarchy	Pass
Arithmetic Over/Under Flows	Pass
Unexpected Ether	Pass
Delegatecall	Pass
Default Public Visibility	Pass
Hidden Malicious Code	Pass
Entropy Illusion (Lack of Randomness)	Pass
External Contract Referencing	Pass
Short Address/ Parameter Attack	Pass
Unchecked CALL Return Values	Pass
Race Conditions / Front Running	Pass
General Denial Of Service (DOS)	Pass
Uninitialized Storage Pointers	Pass
Floating Points and Precision	Pass
Tx.Origin Authentication	Pass
Signatures Replay	Pass
Pool Asset Security (backdoors in the underlying ERC-20)	Pass

We are grateful for the opportunity to work with the Ondo Finance team.

The statements made in this document should not be interpreted as an investment or legal advice, nor should its authors be held accountable for the decisions made based on them.

Zokyo Security recommends the Ondo Finance team put in place a bug bounty program to encourage further analysis of the smart contract by third parties.

zokyo