OPIUM NETWORK TOKEN SMART CONTRACT AUDIT

January 25, 2021

MixBytes()

CONTENTS

1. INTRODUCTION

1.1 DISCLAIMER

The audit makes no statements or warranties about utility of the code, safety of the code, suitability of the business model, investment advice, endorsement of the platform or its products, regulatory regime for the business model, or any other statements about fitness of the contracts to purpose, or their bug free status. The audit documentation is for discussion purposes only. The information presented in this report is confidential and privileged. If you are reading this report, you agree to keep it confidential, not to copy, disclose or disseminate without the agreement of Opium Network. If you are not the intended recipient(s) of this document, please note that any disclosure, copying or dissemination of its content is strictly forbidden.

1.2 PROJECT OVERVIEW

The Opium protocol is a universal protocol to create, settle and trade virtually all derivatives and financial instruments in a professional and trustless way. It allows anyone to build custom exchange-traded products on top of the Ethereum blockchain. Once created, they can be traded freely via a network of relayers and will be priced according to supply and demand.

1.3 SECURITY ASSESSMENT METHODOLOGY

At least 2 auditors are involved in the work on the audit who check the provided source code independently of each other in accordance with the methodology described below:

01 "Blind" audit includes:
> Manual code study
> "Reverse" research and study of the architecture of the code based on the source code only

Stage goal:
Building an independent view of the project's architecture
Finding logical flaws

02 Checking the code against the checklist of known vulnerabilities includes:
> Manual code check for vulnerabilities from the company's internal checklist
> The company's checklist is constantly updated based on the analysis of hacks, research and audit of the clients' code

Stage goal:
Eliminate typical vulnerabilities (e.g. reentrancy, gas limit, flashloan attacks, etc.)

03 Checking the logic, architecture of the security model for compliance with the desired model, which includes:
> Detailed study of the project documentation
> Examining contracts tests
> Examining comments in code
> Comparison of the desired model obtained during the study with the reversed view obtained during the blind audit

Stage goal:
Detection of inconsistencies with the desired model

04 Consolidation of the reports from all auditors into one common interim report document
> Cross check: each auditor reviews the reports of the others
> Discussion of the found issues by the auditors
> Formation of a general (merged) report

Stage goal:
Re-check all the problems for relevance and correctness of the threat level
Provide the client with an interim report

05 Bug fixing & re-check.
> Client fixes or comments on every issue
> Upon completion of the bug fixing, the auditors double-check each fix and set the statuses with a link to the fix

Stage goal:
Preparation of the final code version with all the fixes

06 Preparation of the final audit report and delivery to the customer.

Findings discovered during the audit are classified as follows:

FINDINGS SEVERITY BREAKDOWN

Level	Description	Required action
Critical	Bugs leading to assets theft, fund access locking, or any other loss funds to be transferred to any party	Immediate action to fix issue
Major	Bugs that can trigger a contract failure. Further recovery is possible only by manual modification of the contract state or replacement.	Implement fix as soon as possible
Warning	Bugs that can break the intended contract logic or expose it to DoS attacks	Take into consideration and implement fix in certain period
Comment	Other issues and recommendations reported to/acknowledged by the team	Take into consideration

Based on the feedback received from the Customer's team regarding the list of findings discovered by the Contractor, they are assigned the following statuses:

Status	Description
Fixed	Recommended fixes have been made to the project code and no longer affect its security.
Acknowledged	The project team is aware of this finding. Recommendations for this finding are planned to be resolved in the future. This finding does not affect the overall safety of the project.
No issue	Finding does not affect the overall safety of the project and does not violate the logic of its work.

1.4 EXECUTIVE SUMMARY

The audited scope includes Opium Protocol ERC-20 governance token and lightweight wrapper that covert Opium ERC-20 token to special Opium voting token by 1:1.

1.5 PROJECT DASHBOARD

Client	Opium Network
Audit name	Token
Initial version	188c77ef1e7f693f08e3c74a575143d8ad807787
Final version	ff7385372321cbb481444afdd8d442d82ce64e29
SLOC	43
Date	2021-01-13 - 2021-01-25
Auditors engaged	2 auditors

FILES LISTING

AragonVotingWrapper.sol	AragonVotingWrapper.sol
OpiumToken.sol	OpiumToken.sol

FINDINGS SUMMARY

Level	Amount
Critical	0
Major	0
Warning	1
Comment	1

CONCLUSION

Smart contracts were audited and no critical and major issues were found. During audit two minor issues were spotted and marked with warning and comment level which were fixed or acknowledged by client while working on report results. So, the contracts are assumed as secure to use according to our security criteria.

2. FINDINGS REPORT

2.1 CRITICAL

Not Found

2.2 MAJOR

Not Found

2.3 WARNING

WRN-1	Potential inability to wrap/unwrap
File	AragonVotingWrapper.sol
Severity	Warning
Status	Acknowledged

DESCRIPTION

Due to the wrap/unwrap functionality based on token in/out transfers in the `AragonVotingWrapper` contract located at AragonVotingWrapper.sol there are implicit requirements: both governance and voting tokens should have the same total supply and the wrapper contract should always have a sufficient amount of both tokens to cover all potential wrap/unwrap requests. For now, that invariant is never checked in the contract code and the contract can potentially reach a state when it won't be able to serve a wrap/unwrap operation.

RECOMMENDATION

We recommend to use wrap/unwrap mechanic using mint/burn. This approach allows to avoid situations with an insufficient balance.

CLIENT'S COMMENTARY

> This is a problem with the initial setup, we will take it into account. We will not burn and mint tokens, so we will crush OPIUM and VOPIUM of the same 100M each and make sure with the initial setup that OPIUM + VOPIUM = 100M are in the wrapper.

2.4 COMMENTS

CMT-1	Unused pragma declaration
File	AragonVotingWrapper.sol OpiumToken.sol
Severity	Comment
Status	Fixed at a1d6ff93

DESCRIPTION

At lines AragonVotingWrapper.sol#L4 and OpiumToken.sol#L4 there is a declared pragma that enables an experimental ABI encoder. However, contracts never use features from that encoder.

RECOMMENDATION

We suggest removing the unused declaration.

3. ABOUT MIXBYTES

MixBytes is a team of blockchain developers, auditors and analysts keen on decentralized systems. We build open-source solutions, smart contracts and blockchain protocols, perform security audits, work on benchmarking and software testing solutions, do research and tech consultancy.

BLOCKCHAINS

 Ethereum

 Cosmos

 EOS

 Substrate

TECH STACK

 Python

 Solidity

 Rust

 C++

CONTACTS

 https://github.com/mixbytes/audits_public

 https://mixbytes.io/

 hello@mixbytes.io

 https://t.me/MixBytes

 https://twitter.com/mixbytes