CONTRACT WOLF

Security Assessment

Oracle AI

Verified on 04/20/2023

SUMMARY

Project	CHAIN	METHODOLOGY
Oracle AI	Arbitrum	Manual & Automatic Analysis

FILES	DELIVERY	TYPE
Single	04/20/2023	Standard Audit

4	0	1	0	0	3
Total Findings	Critical	Major	Medium	Minor	Informational

🟥 0 Critical	0 Pending	An exposure that can affect the contract functions in several events that can risk and disrupt the contract
🟧 1 Major	1 Pending	An exposure that can affect the outcome when using the contract that can serve as an opening in manipulating the contract in an unwanted manner
🟧 0 Medium	0 Pending	An opening that could affect the outcome in executing the contract in a specific situation
⬜ 0 Minor	0 Pending	An opening but doesn't have an impact on the functionality of the contract
🟦 3 Informational	3 Pending	An opening that consists information but will not risk or affect the contract
STATUS	✔️ **AUDIT PASSED**	

TABLE OF CONTENTS | Oracle AI

DISCLAIMER | Oracle AI

ContractWolf audits and reports should not be considered as a form of project's "Advertisement" and does not cover any interaction and assessment from "Project Contract" to "External Contracts" such as PancakeSwap, UniSwap, SushiSwap or similar.

ContractWolf does not provide any warranty on its released report and should not be used as a decision to invest into audited projects.

ContractWolf provides a transparent report to all its "Clients" and to its "Clients Participants" and will not claim any guarantee of bug-free code within its **SMART CONTRACT**.

ContractWolf's presence is to analyze, audit and assess the Client's Smart Contract to find any underlying risk and to eliminate any logic and flow errors within its code.

Each company or project should be liable to its security flaws and functionalities.

SCOPE OF WORK | Oracle AI

Oracle AI team has agreed and provided us with the files that need to be tested (*Github, BSCscan, Etherscan, Local files etc*). The scope of audit is the main contract.

The goal of this engagement is to identify if there is a possibility of security flaws in the implementation of smart contract and its systems.

ContractWolf will be focusing on contract issues and functionalities along with the project claims from smart contract to their website, whitepaper, repository which has been provided by **Oracle AI**.

AUDITING APPROACH | Oracle AI

Every line of code along with its functionalities will undergo manual review to check for security issues, quality of logic and contract scope of inheritance. The manual review will be done by our team that will document any issues that they discovered.

METHODOLOGY

The auditing process follows a routine series of steps :

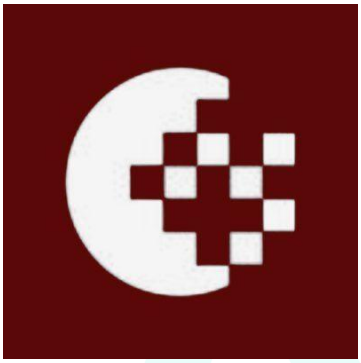
1. Code review that includes the following :
   - Review of the specifications, sources and instructions provided to ContractWolf to make sure we understand the size, scope and functionality of the smart contract.
   - Manual review of code. Our team will have a process of reading the code line-by-line with the intention of identifying potential vulnerabilities, underlying and hidden security flaws.

2. Testing and automated analysis that includes :
   - Testing the smart contract function with common test cases and scenarios to ensure that it returns the expected results.

3. Best practices and ethical review. The team will review the contract with the aim to improve efficiency, effectiveness, clarifications, maintainability, security and control within the smart contract.

4. Recommendations to help the project take steps to eliminate or minimize threats and secure the smart contract.

TOKEN DETAILS | Oracle AI



Oracle AI is an invaluable tool for investors and traders seeking to make informed decisions and capitalize on new opportunities before others even know they exist.

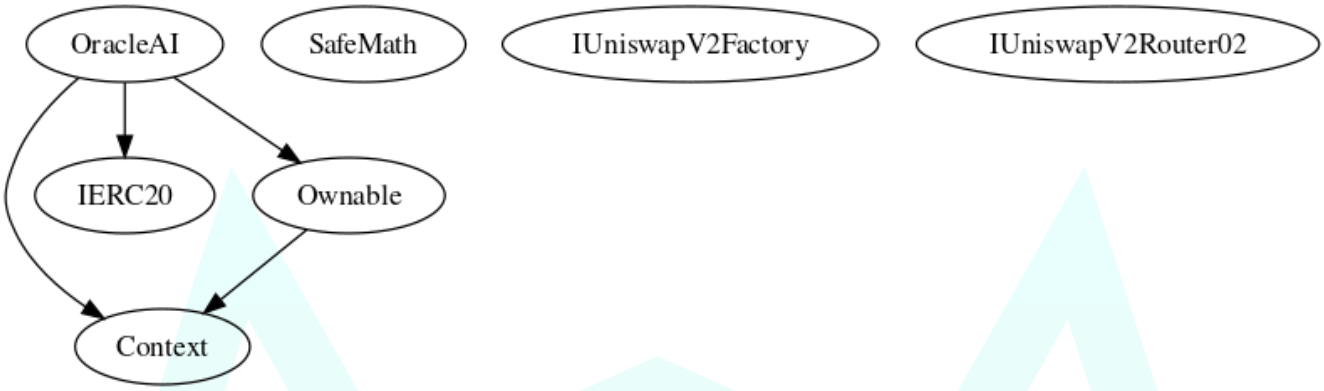
Token Name	Symbol	Decimal	Total Supply	Chain
Oracle AI	ORCAI	6	1,000,000	Arbitrum

SOURCE

Source	*Sent Via local-files*

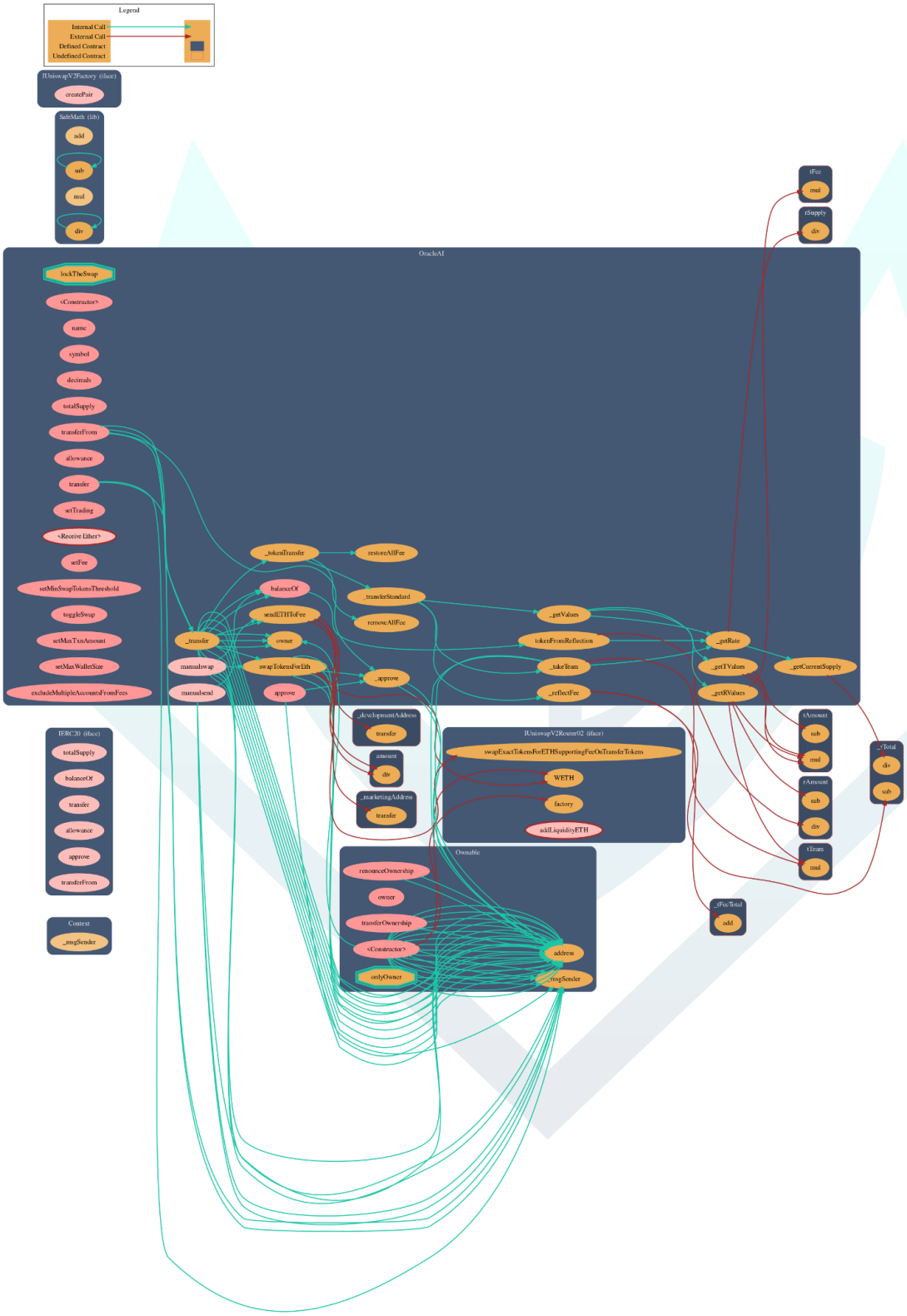
INHERITANCE GRAPH | Oracle AI

Inheritance Graph of Contract Functions



CALL GRAPH | Oracle AI

Call Graph of Contract Functions



FINDINGS | Oracle AI

4	0	1	0	0	3
Total Findings	Critical	Major	Medium	Minor	Informational

This report has been prepared to state the issues and vulnerabilities for Oracle AI through this audit. The goal of this report findings is to identify specifically and fix any underlying issues and errors

ID	Title	File & Line #	Severity	Status
SWC-107	Reentrancy Attack	OracleAI.sol, L: 347	Major	• Pending
SWC-103	Floating Pragma is set	OracleAI.sol, L: 11	Informational	• Pending
SWC-131	Presence of Unused Variable	OracleAI.sol, L: 174, 199	Informational	• Pending
CW-102	SafeMath Override	OracleAI.sol, L: 88	Informational	• Pending

SWC ATTACKS | Oracle AI

Smart Contract Weakness Classification and Test Cases

ID	Description	Status
SWC-100	Function Default Visibility	Passed
SWC-101	Integer Overflow and Underflow	Passed
SWC-102	Outdated Compiler Version	Passed
SWC-103	Floating Pragma	Not Passed
SWC-104	Unchecked Call Return Value	Passed
SWC-105	Unprotected Ether Withdrawal	Passed
SWC-106	Unprotected SELF DESTRUCT Instruction	Passed
SWC-107	Reentrancy	Not Passed
SWC-108	State Variable Default Visibility	Passed
SWC-109	Uninitialized Storage Pointer	Passed
SWC-110	Assert Violation	Passed
SWC-111	Use of Deprecated Solidity Functions	Passed
SWC-112	Delegatecall to Untrusted Callee	Passed
SWC-113	DoS with Failed Call	Passed
SWC-114	Transaction Order Dependence	Passed
SWC-115	Authorization through tx.origin	Passed
SWC-116	Block values as a proxy for time	Passed
SWC-117	Signature Malleability	Passed
SWC-118	Incorrect Constructor Name	Passed
SWC-119	Shadowing State Variables	Passed
SWC-120	Weak Sources of Randomness from Chain Attributes	Passed
SWC-121	Missing Protection against Signature Replay Attacks	Passed
SWC-122	Lack of Proper Signature Verification	Passed

ID	Description	Status
SWC-123	Requirement Violation	Passed
SWC-124	Write to Arbitrary Storage Location	Passed
SWC-125	Incorrect Inheritance Order	Passed
SWC-126	Insufficient Gas Griefing	Passed
SWC-127	Arbitrary Jump with Function Type Variable	Passed
SWC-128	DoS With Block Gas Limit	Passed
SWC-129	Typographical Error	Passed
SWC-130	Right-To-Left-Override control character(U+202E)	Passed
SWC-131	Presence of unused variables	Not Passed
SWC-132	Unexpected Ether balance	Passed
SWC-133	Hash Collisions With Multiple Variable Arguments	Passed
SWC-134	Message call with hardcoded gas amount	Passed
SWC-135	Code With No Effects	Passed
SWC-136	Unencrypted Private Data On-Chain	Passed

CW ASSESSMENT | Oracle AI

ContractWolf Vulnerability and Security Tests

ID	Name	Description	Status
CW-001	Multiple Version	Presence of multiple compiler version across all contracts	✔
CW-002	Incorrect Access Control	Additional checks for critical logic and flow	✔
CW-003	Payable Contract	A function to withdraw ether should exist otherwise the ether will be trapped	✔
CW-004	Custom Modifier	major recheck for custom modifier logic	✔
CW-005	Divide Before Multiply	Performing multiplication before division is generally better to avoid loss of precision	✔
CW-006	Multiple Calls	Functions with multiple internal calls	✔
CW-007	Deprecated Keywords	Use of deprecated functions/operators such as block.blockhash() for blockhash(), msg.gas for gasleft(), throw for revert(), sha3() for keccak256(), callcode() for delegatecall(), suicide() for selfdestruct(), constant for view or var for actual type name should be avoided to prevent unintended errors with newer compiler versions	✔
CW-008	Unused Contract	Presence of an unused, unimported or uncalled contract	✔
CW-009	Assembly Usage	Use of EVM assembly is error-prone and should be avoided or double-checked for correctness	✔
CW-010	Similar Variable Names	Variables with similar names could be confused for each other and therefore should be avoided	✔
CW-011	Commented Code	Removal of commented/unused code lines	✔
CW-012	SafeMath Override	SafeMath is no longer needed starting Solidity v0.8+. The compiler now has Built in overflow checking.	✘

FIXES & RECOMMENDATION

SWC-103 | A Floating Pragma is Set

Code

```
// SPDX-License-Identifier: UNLICENSED
pragma solidity ^0.8.4;
```

The compiler version should be a fixed one to avoid undiscovered compiler bugs. Fixed version sample below

```
// SPDX-License-Identifier: UNLICENSED
pragma solidity 0.8.18;
```

SWC-107 | Reentrancy

Functions `_transfer` Reentrancy detected through external calls.

Recommendation

Make sure all internal state changes are performed before the call is executed or;
Use Openzeppelin's ReentrancyGuard

SWC-131 | Presence of unused variable

Although unused variables are allowed in Solidity and may not directly pose a security risk, it is still recommended to avoid them. This is because they can cause unnecessary gas consumption, suggest poor code quality and indicate bugs or malformed data structures. Additionally, they can make the code less readable and create unnecessary noise.

```
mapping(address => uint256) private _tOwned;
mapping(address => uint256) private cooldown;
```

Recommendation

Remove all unused variables from the code base.

CW-012 | SafeMath Override

```
library SafeMath
```

SafeMath is no longer needed starting Solidity v0.8+. The compiler now has Built in overflow checking.

Max transaction amount and wallet amount can be set to zero can be used to pause transactions

Owner can update max transaction amount and max wallet amount with an indefinite amount

```
    function setMaxTxnAmount(uint256 maxTxAmount) public onlyOwner {
        _maxTxAmount = maxTxAmount;
    }

    function setMaxWalletSize(uint256 maxWalletSize) public onlyOwner {
        _maxWalletSize = maxWalletSize;
    }
```

Recommendation

Adding validation checks to the maxTransactionAmount and maxWallet parameters to ensure they are not equal to zero and are greater than zero can prevent unintended behavior or error.

AUDIT COMMENTS | Oracle AI

Smart Contract audit comment for a non-technical perspective

🟢 Owner can renounce and transfer ownership
🟢 Owner can toggle trading
🟢 Owner can update both total fees on buy and sell not greater than 20% each
🟢 Owner can update minimum tokens for swap with an indefinite amount
🟢 Owner can toggle swap
🟢 Owner can update max transaction with an indefinite amount
🟢 Owner can update max wallet with an indefinite amount
🟢 Owner can exclude/include addresses from fees
🟢 Owner cannot burn tokens
🟢 Owner cannot mint after initial deployment
🟢 Owner cannot block users

CONTRACTWOLF

Blockchain Security - Smart Contract Audits