CONTRACT WOLF

Blockchain Security - Smart Contract Audits

Security Assessment

March 22, 2023

Disclaimer

ContractWolf.io audits and reports should not be considered as a form of project's "advertisement" and does not cover any interaction and assessment from "project's contract" to "external contracts" such as Pancakeswap or similar.

ContractWolf does not provide any warranty on its released reports.

ContractWolf should not be used as a decision to invest into an audited project and is not affiliated nor partners to its audited contract projects.

ContractWolf provides 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 presence is to analyze, audit and assess the client's smart contract's code.

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

Scope of Work

Artificial Intel's team agreed and provided us with the files that needs to be tested (Github, Bscscan, Etherscan, files, etc.). The scope of the audit is the main contract.

The goal of this engagement was to identify if there is a possibility of security flaws in the implementation of the contract or system.

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

Description

The world's first AI-developed and managed web3 project. AI building AI. Led by Butcher the AI dev, Artificial Intel will creaet Arbitrum's first synthetic Gas Futures contracts that allow a user to speculate on gas prices across multiple chains. This project is very much an experiment of how well an AI can develop and make managerial decisions to further a web3 project. Beginning with a tradeable marketplace of futures, Butcher will endeavor to become the first successful AI dev on the planet!

Risk Level Classification

Risk Level represents the classification or the probability that a certain function or threat that can exploit vulnerability and have an impact within the system or contract.
Risk Level is computed based on CVSS Version 3.0

Level	Value	Vulnerability
Critical	9 - 10	An Exposure that can affect the contract functions in several events that can risk and disrupt the contract
High	7 - 8.9	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
Medium	4 - 6.9	An opening that could affect the outcome in executing the contract in a specific situation
Low	0.1 - 3.9	An opening but doesn't have an impact on the functionality of the contract
Informational	0	An opening that consists of information's but will not risk or affect the contract

Auditing Approach

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

2. Testing and automated analysis that includes:

    - Testing the smart contract functions with common test cases and scenarios, to ensure that it returns the expected results.

3. Best practices 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 secure the smart contract.

Used Code from other Frameworks/Smart Contracts (Direct Imports)

Imported Packages

- Address
- Aritificial_IntelV1
- Context
- ERC20
- IERC20
- IERC20Metadata
- IERC20Permit
- IUniswapV2Caller
- IUniswapV2Factory
- IUniswapV2Pair
- IUniswapV2Router01
- IUniswapV2Router02
- Ownable
- SafeERC

Description

Optimization enabled: Yes

Decimal: 18

Symbol: AI1

Max / Total Supply: 1,000,000

Capabilities

Components

Version	Contracts	Libraries	Interfaces	Abstract
1.0	2	2	8	2

Exposed Functions

Version	Public	Private	External	Internal
1.0	16	6	60	31

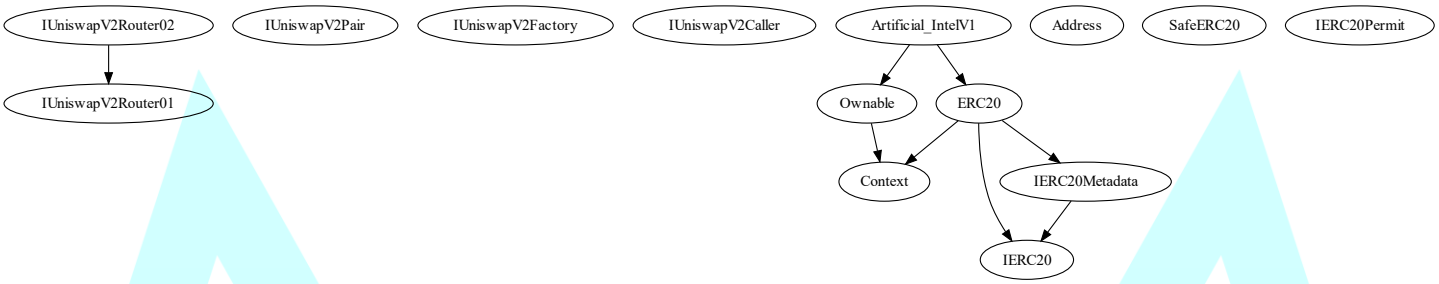
State Variables

Version	Total	Public
1.0	26	16

Capabilities

Version	Solidity Versions Observed	Experimental Features	Can Receive Funds	Uses Assembly	Has Destroyable Contracts
1.0	v0.8.13		Yes	Yes	No

Inheritance Graph



Correct implementation of Token Standard

Tested	Verified
✓	✓

Overall Checkup (Smart Contract Security)

Tested	Verified
✓	✓

Function	Description	Exist	Tested	Verified
TotalSupply	Information about the total coin or token supply	✓	✓	✓
BalanceOf	Details on the account balance from a specified address	✓	✓	✓
Transfer	An action that transfers a specified amount of coin or token to a specified address	✓	✓	✓
TransferFrom	An action that transfers a specified amount of coin or token from a specified address	✓	✓	✓
Approve	Provides permission to withdraw specified number of coin or token from a specified address	✓	✓	✓

Verify Claims

Statement	Exist	Tested	Deployer
Renounce Ownership	✓	✓	✓
Mint	✓	✓	✗
Burn	✓	✓	✗
Block	–	–	–
Pause	–	–	–

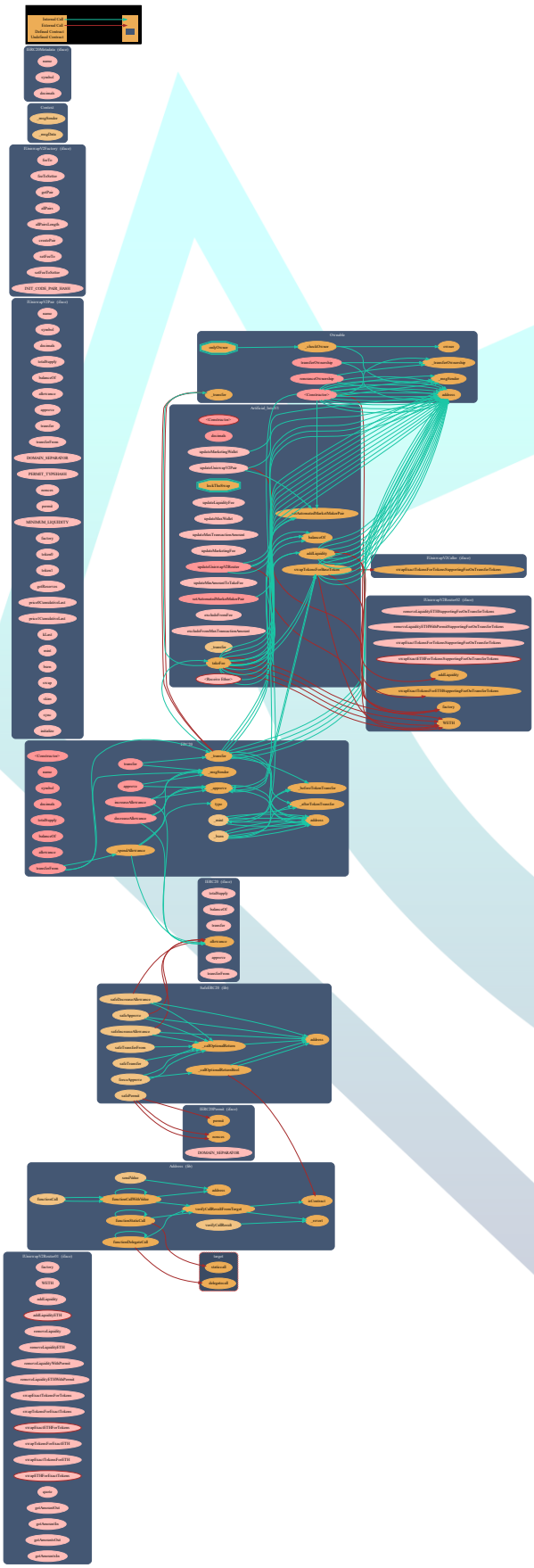
Legend

Attribute	Symbol
Verified / Can	✓
Verified / Cannot	✗
Unverified / Not checked	⚑
Not Available	–

Write Functions of Contract

1. approve (0x095ea7b3)
2. decreaseAllowance (0xa457c2d7)
3. excludeFromFee (0xdf8408fe)
4. excludeFromMaxTransactionAmount (0x2ae2f121)
5. increaseAllowance (0x39509351)
6. renounceOwnership (0x715018a6)
7. setAutomatedMarketMakerPair (0x9a7a23d6)
8. transfer (0xa9059cbb)
9. transferFrom (0x23b872dd)
10. transferOwnership (0xf2fde38b)
11. updateLiquidityFee (0xd68f8cde)
12. updateMarketingFee (0xcf089e13)
13. updateMarketingWallet (0x4707c551)
14. updateMaxTransactionAmount (0xaa498023)
15. updateMaxWallet (0x1c499ab0)
16. updateMinAmountToTakeFee (0x73b9e82c)
17. updateUniswapV2Pair (0x91c1004a)
18. updateUniswapV2Router (0x65b8dbc0)

Call Graph


External Call
Internal Call
Defined Contract
Undefined Contract
IERC20Metadata (iface)
name
symbol
decimals
Context
_msgSender
_msgData
IUniswapV2Factory (iface)
feeTo
feeToSetter
getPair
allPairs
allPairsLength
createPair
setFeeTo
setFeeToSetter
INIT_CODE_PAIR_HASH
IUniswapV2Pair (iface)
name
symbol
decimals
totalSupply
balanceOf
allowance
approve
transfer
transferFrom
DOMAIN_SEPARATOR
PERMIT_TYPEHASH
nonces
permit
MINIMUM_LIQUIDITY
factory
token0
token1
getReserves
price0CumulativeLast
price1CumulativeLast
kLast
mint
burn
swap
skim
sync
initialize
Ownable
owner
_checkOwner
renounceOwnership
transferOwnership
_transferOwnership
_msgSender
ArtificialLiquidityV2
updateMarketingWallet
updateLiquidityFee
updateMaxWallet
updateMaxTransactionAmount
updateMarketingFee
updateLiquidityRatio
updateMaxAmountTxTakeFee
setAutomatedMarketMakerPair
excludeFromFee
includeFromTransactionAmount
balanceOf
addLiquidity
swapTokensForEthereum
IUniswapV2Router02 (iface)
removeLiquidityETHSupportingFeeOnTransferTokens
swapExactTokensForTokensSupportingFeeOnTransferTokens
swapExactETHForTokensSupportingFeeOnTransferTokens
factory
WETH
IERC20
totalSupply
balanceOf
transfer
allowance
approve
transferFrom
SafeERC20 (lib)
safeDecreaseAllowance
safeApprove
safeIncreaseAllowance
safeTransferFrom
safeTransfer
_callOptionalReturn
forceApprove
safePermit
IERC20Permit (iface)
permit
nonces
DOMAIN_SEPARATOR
Address (lib)
sendValue
functionCall
functionCallWithValue
isContract
functionStaticCall
verifyCallResult
functionDelegateCall
target
staticcall
delegatecall
IUniswapV2Router02 (iface)
factory
WETH
addLiquidity
addLiquidityETH
removeLiquidity
removeLiquidityETH
removeLiquidityWithPermit
removeLiquidityETHWithPermit
swapExactTokensForTokens
swapTokensForExactTokens
swapExactETHForTokens
swapTokensForExactETH
swapExactTokensForETH
swapETHForExactTokens
quote
getAmountOut
getAmountIn
getAmountsOut
getAmountsIn

SWC Attacks

ID	Title	Status
SWC-136	Unencrypted Private Data On-Chain	PASSED
SWC-135	Code With No Effects	PASSED
SWC-134	Message call with hardcoded gas amount	PASSED
SWC-133	Hash Collisions with Multiple Variable Length Arguments	PASSED
SWC-132	Unexpected Ether balance	PASSED
SWC-131	Presence of unused variables	PASSED
SWC-130	Right-To Left Override control character (U+202E)	PASSED
SWC-129	Typographical Error	PASSED
SWC-128	DoS With Block Gas Limit	PASSED
SWC-127	Arbitrary Jump with Function Type Variable	PASSED
SWC-126	Insufficient Gas Griefing	PASSED
SWC-125	Incorrect Inheritance Order	PASSED
SWC-124	Write to Arbitrary Storage Location	PASSED
SWC-123	Requirement Violation	PASSED
SWC-122	Lack of Proper Signature Verification	PASSED
SWC-121	Missing Protection against Signature Replay Attacks	PASSED
SWC-120	Weak Sources of Randomness from Chain Attributes	PASSED
SWC-119	Shadowing State Variables	PASSED
SWC-118	Incorrect Constructor Name	PASSED
SWC-117	Signature Malleability	PASSED
SWC-116	Block values as a proxy for time	PASSED
SWC-115	Authorization through tx.origin	PASSED
SWC-114	Transaction Order Dependence	PASSED
SWC-113	DoS with Failed Call	PASSED
SWC-112	Delegate call to Untrusted Callee	PASSED
SWC-111	Use of Deprecated Solidity Functions	PASSED

SWC-110	Assert Violation	**PASSED**
SWC-109	Uninitialized Storage Pointer	**PASSED**
SWC-108	State Variable Default Visibility	**PASSED**
SWC-107	Reentrancy	**PASSED**
SWC-106	Unprotected SELFDESTRUCT Instruction	**PASSED**
SWC-105	Unprotected Ether Withdrawal	**PASSED**
SWC-104	Unchecked Call Return Value	**PASSED**
SWC-103	Floating Pragma	**PASSED**
SWC-102	Outdated Compiler Version	**PASSED**
SWC-101	Integer Overflow and Underflow	**PASSED**
SWC-100	Function Default Visibility	**PASSED**

Audit Result

AUDIT PASSED

Critical Issues

No critical issues found

High Issues

No high issues found

Medium Issues

No medium issues found

Low Issues

No low issues found

Informational Issues

No informational issues found

Function Issues

No informational issues found

Findings

Owner can set max tx limit

```
function updateMaxTransactionAmount(uint256 _maxTransactionAmount)
    external
    onlyOwner
{
    require(_maxTransactionAmount>0, "maxTransactionAmount > 0");
    emit UpdateMaxTransactionAmount(_maxTransactionAmount, maxTransactionAmount);
    maxTransactionAmount = _maxTransactionAmount;
}
```

Owner can set max wallet limit

```
function updateMaxWallet(uint256 _maxWallet) external onlyOwner {
    require(_maxWallet>0, "maxWallet > 0");
    emit UpdateMaxWallet(_maxWallet, maxWallet);
    maxWallet = _maxWallet;
}
```

Owner can set total fees up to 20%

```
function updateMarketingFee(
    uint16 _sellMarketingFee,
    uint16 _buyMarketingFee
) external onlyOwner {
    require(
        _sellMarketingFee + (sellLiquidityFee) <= 200,
        "sell fee <= 20%"
    );
    require(_buyMarketingFee + (buyLiquidityFee) <= 200, "buy fee <= 20%");
    emit UpdateMarketingFee(
        _sellMarketingFee,
        _buyMarketingFee,
        sellMarketingFee,
        buyMarketingFee
    );
    sellMarketingFee = _sellMarketingFee;
    buyMarketingFee = _buyMarketingFee;
}
```

Function Comments

- Owner can renounce ownership
- Owner can transfer ownership
- Owner can update uniswap v2 pair address
- Owner can update uniswap v2 router address
- Owner can set total fees up to 20%
- Owner can set max wallet limit not lower than 0
- Owner can set max tx limit not lower than 0
- Owner can update marketing wallet receiver address
- Owner can update minimum amount to take fee
- Owner can toggle and set automated market maker pair address
- Owner can include/exclude address from fees
- Owner can include/exclude address from max tx limit
- Owner cannot burn tokens
- Owner cannot mint after initial deployment
- Owner cannot pause contract
- Owner cannot block user

CONTRACTWOLF

Blockchain Security - Smart Contract Audits