

SKELETON ECOSYSTEM

SMART CONTRACT AUDIT



Aura AI
\$AURA
ERC20

0x5d19d39852bf99bc4344483cbce3c9ff8f026b43D06



Table of Contents

Table of Contents	1
Disclaimer	2
Overview	3
Creation/Audit Date	3
Verified Socials	3
Contract Functions Analysis	4
Contract Safety and Weakness	8
Detected Vulnerability Description	12
Contract Flow Graph	14
Contract Interaction Graph	15
Inheritance Graph	16
Contract Descriptions	17
Audit Scope	20

Global Disclaimer

This document serves as a disclaimer for the crypto smart contract audit conducted by Skeleton Ecosystem. The purpose of the audit was to review the codebase of the smart contracts for potential vulnerabilities and issues. It is important to note the following:

Limited Scope: The audit is based on the code and information available up to the audit completion date. It does not cover external factors, system interactions, or changes made after the audit. The audit itself can not guarantee 100% safety and can not detect common scam methods like farming and developer sell-out.

No Guarantee of Security: While we have taken reasonable steps to identify vulnerabilities, it is impossible to guarantee the complete absence of security risks or issues. The audit report provides an assessment of the contract's security as of the audit date.

Continued Development: Smart contracts and blockchain technology are evolving fields. Updates, forks, or changes to the contract post-audit may introduce new risks that were not present during the audit.

Third-party Code: If the smart contract relies on third-party libraries or code, those components were not thoroughly audited unless explicitly stated. Security of these dependencies is the responsibility of their respective developers.

Non-Exhaustive Testing: The audit involved automated analysis, manual review, and testing under controlled conditions. It is possible that certain vulnerabilities or issues may not have been identified.

Risk Evaluation: The audit report includes a risk assessment for identified vulnerabilities. It is recommended that the development team carefully reviews and addresses these risks to mitigate potential exploits.

Not Financial Advice: This audit report is not intended as financial or investment advice. Decisions regarding the use, deployment, or investment in the smart contract should be made based on a comprehensive assessment of the associated risks.

By accessing and using this audit report, you acknowledge and agree to the limitations outlined above. Skeleton Ecosystem and its auditors shall not be held liable for any direct or indirect damages resulting from the use of the audit report or the smart contract itself.

Please consult with legal, technical, and financial professionals before making any decisions related to the smart contract.

Overview

Contract Name	AURA
Ticker/Symbol	\$AURA
Blockchain	Ethereum ERC20
Contract Address	0x5d19d39852bf99bc4344483cbce3c9ff8f026b43
Creator Address	0xDfDd0Df6841fF71F4340E6Dc755CC61c06daf59e
Current Owner Address	0x00
Contract Explorer	https://etherscan.io/address/0x5d19d39852bf99bc4344483cbce3c9ff8f026b43#code
Compiler Version	v0.8.23+commit.f704f362
License	MIT
Optimisation	No with 200 Runs
Total Supply	100,000,000 AURA
Decimals	9

Creation/Audit

Contract Deployed	17.12.2024
Audit Created	19.12.2024
Audit Update	V 1.0

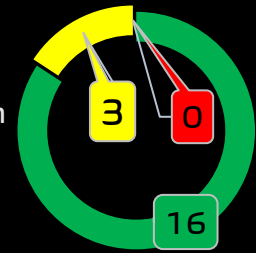
Verified Socials








Website	https://aura-ai.xyz/
Telegram	https://t.me/auraaierc
Twitter (X)	https://x.com/auraaierc

Contract Function Analysis

 Pass
  Attention Item
  Risky Item

■ Pass
 ■ Attention
 ■ Risk







Contract Verified		The contract source code is uploaded to blockchain explorer and is open source, so everybody can read it.
Contract Ownership		0x00
Buy Tax	5 %	Shows the taxes for purchase transactions. Above 10% may be considered a high tax rate. More than 50% tax rate means may not be tradable. Fee can be set!
Sell Tax	5 %	Shows the taxes for sell transactions. Above 10% may be considered a high tax rate. More than 50% tax rate means may not be tradable. Fee can be set!
Honeypot Analyse		Holder is able to buy and sell. If honeypot: The contract blocks sell transfer from holder wallet. Multiple events may cause honeypot. Trading disabled, extremely high tax
Liquidity Status		Liquidity status on 19.12.2024 Locked 99% for 363 days on Unicrypt https://etherscan.io/tx/0x4018e461303c436b40950703495570a7f4bb7154999f9c314523cd960161ddde
Trading Disable Functions		No Trading suspendable function found. If a suspendable code is included, the token maybe neither be bought or sold (honeypot risk). If contract is renounced this function can't be used
Set Fees function	 max 5%	Fee Setting function found. Contract renounced, function can not be triggered by owner. The contract owner may contain the authority to modify the transaction tax. If the transaction tax is increased to more than 49%, the tokens may not be able to be traded (honeypot risk).
Proxy Contract		Not a Proxy contract
Mint Function		No Mint Function detected Mint function is transparent or non-existent. Hidden mint functions may increase the amount of tokens in circulation and effect the price of the token. Owner can mint new tokens and sell.

Balance Modifier Function		<p>No Balance Modifier function found.</p> <p>If there is a function for this, the contract owner can have the authority to modify the balance of tokens at other addresses. For example revoke the bought tokens from the holders wallet. Common form of scam: You buy the token, but it's disappearing from your wallet.</p>
Blacklist Function		<p>No Blacklist Setting function found.</p> <p>If there is a blacklist, some addresses may not be able to trade normally. Example: you buy the token and right after your Wallet getting blacklisted. Like so you will be unable to sell. Honeypot Risk.</p>
Whitelist Function		<p>No Whitelist Setting function found.</p> <p>If there is a function for this Developer can set zero fee or no max wallet size for addresses (for example team wallets can trade without fee. Can cause farming)</p>
Hidden Owner Analysis		<p>No Hidden or multi owner with authorisation</p> <p>For contract with a hidden owner, developer can still manipulate the contract even if the ownership has been abandoned.</p>
Retrieve Ownership Function		<p>No Functions found which can retrieve ownership of the contract.</p> <p>If this function exists, it is possible for the project owner to regain ownership even after relinquishing it. Also known as fake renounce.</p>
Self Destruct Function		<p>No Self Destruct function found.</p> <p>If this function exists and is triggered, the contract will be destroyed, all functions will be unavailable, and all related assets will be erased.</p>
Transfer Tax Changing Function		<p>Transfer Tax Changing Functions found.</p> <p>Contract renounced, function can not be triggered by owner.</p> <p>If it exists, the contract owner may set a very outrageous tax rate for assigned address to block it from trading. Can assign all wallets at once!</p>
Trading Cooldown Function		<p>No Trading Cooldown Function found. If there is a trading cooldown function, the user will not be able to sell the token within a certain time or block after buying. Like a temporary honeypot.</p>
Max Transaction and Holding Modify Function		<p>Max Transaction and Holding Modify function found. Contract renounced, function can not be triggered by owner.</p> <p>If there is a function for this, the maximum trading amount or maximum position can be modified. Can cause honeypot</p>
Transaction Limiting Function		<p>No Transaction Limiter Function Found.</p> <p>The number of overall token transactions may be limited (honeypot risk)</p>

Details of Risk - Attention Items

Removing Risk of contract function based on renounced ownership

Transaction Receipt Event Logs

158 **Address** 0x5d19d39852bf99bc4344483cbce3c9ff8f026b43    

Name OwnershipTransferred (index_topic_1 address previousOwner, index_topic_2 address newOwner) [View Source](#)

Topics 0 0x8be0079c531659141344cd1fd0a4f28419497f9722a3daafe3b4186f6b6457e0

1: previousOwner Dec ▾ → 0xDfd0Df6841fF71F4340E6Dc755CC61c06daf59e

2: newOwner Dec ▾ → 0x00

Data 0x

Following detected contract functions serve as informational purposes about the contract. The owner has no more authorisation to trigger the following functions.

Set Fee (Max 5% limit found)

Contract renounced, function can not be triggered by owner.

The contract owner may contain the authority to modify the transaction tax. If the transaction tax is increased to more than 49%, the tokens may not be able to be traded (honeypot risk).

```

332
ftrace | funcSig
333 function updateFee(uint256 _newFee!) external{
334     require(_msgSender()==_taxWallet);
335     require(_newFee!<=_finalBuyTax && _newFee!<=_finalSellTax);
336     _finalBuyTax=_newFee!;
337     _finalSellTax=_newFee!;
338 }
339

```

```

128
129 uint256 private _initialBuyTax=20;
130 uint256 private _initialSellTax=23;
131 uint256 private _finalBuyTax=5;
132 uint256 private _finalSellTax=5;
133 uint256 private _reducebuyTaxAt=16;
134 uint256 private _reducesellTaxAt=16;
135 uint256 private _preventSwapBefore=26;
136 uint256 private _transferTax=0;
137 uint256 private _buyCount=0;

```

⚠ Transfer Tax changing function

Contract renounced, function can not be triggered by owner.

If it exists, the contract owner may set a very outrageous tax rate for assigned address to block it from trading. Can assign all wallets at once!

```
fttrace | funcSig
296     function updateTransferTax() external onlyOwner{
297         _transferTax = 0;
298         emit TransferTaxUpdated(0);
299     }
300
```

⚠ Max Transaction and Holding Modify function

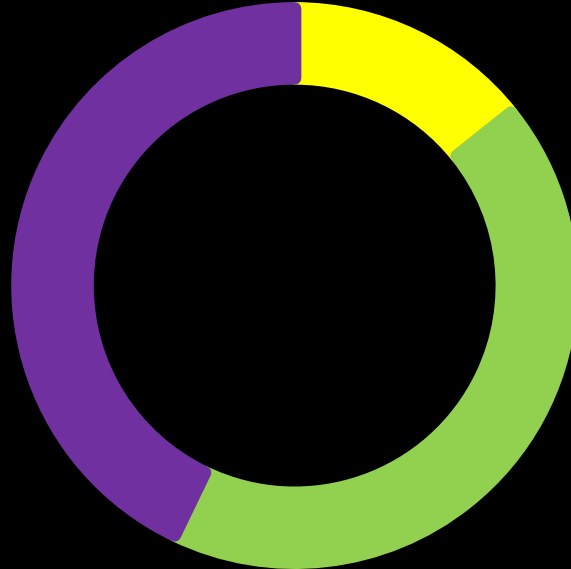
If there is a function for this, the maximum trading amount or maximum position can be modified. Can cause honeypot

Max txn and wallet limit updated before renouncing ownership. limit = total supply

```
fttrace | funcSig
290     function updateLimits() external onlyOwner{
291         _maxTxAmount = _tTotal;
292         _maxWalletSize = _tTotal;
293         emit MaxTxAmountUpdated(_tTotal);
294     }
```


Contract Security

Total Findings: 7



■ **High Severity Issues:** High possibility to cause problems, need to be resolved.

■ **Medium Severity Issue:** Will likely cause problems, recommended to resolve.

■ **Low Severity Issues:** Won't cause problems, but for improvement purposes could be adjusted.

■ **Informational Severity Issues:** Not harmful in any way, information for the developer team.

Contract Security

List of Found Issues

High severity Issues: (0)

Medium severity issues: (1)

- Incorrect Access Control

Low severity issues: (3)

- Missing Events
- Long number literals
- Outdated compiler Version

Informational severity issues: (3)

- Public Functions Should be Declared External
- Costly Loop Operations
- State Variables Should be Declared Constant

Contract Weakness Classification

THE SMART CONTRACT WEAKNESS CLASSIFICATION REGISTRY (SWC REGISTRY) IS AN IMPLEMENTATION OF THE WEAKNESS CLASSIFICATION SCHEME PROPOSED IN EIP-1470. IT IS LOOSELY ALIGNED TO THE TERMINOLOGIES AND STRUCTURE USED IN THE COMMON WEAKNESS ENUMERATION (CWE) WHILE OVERLAYING A WIDE RANGE OF WEAKNESS VARIANTS THAT ARE

ID	Description	AI	Manual	Result
SWC-100	Function Default Visibility	Passed	Passed	Passed
SWC-101	Integer Overflow and Underflow	Passed	Passed	Passed
SWC-102	Outdated Compiler Version	low	low	low
SWC-103	Floating Pragma	Passed	Passed	Passed
SWC-104	Unchecked Call Return Value	Passed	Passed	Passed
SWC-105	Unprotected Ether Withdrawal	Passed	Passed	Passed
SWC-106	Unprotected SELFDESTRUCT Instruction	Passed	Passed	Passed
SWC-107	Reentrancy	Passed	Passed	Passed
SWC-108	State Variable Default Visibility	Passed	Passed	Passed
SWC-109	Uninitialized Storage Pointer	Passed	Passed	Passed
SWC-110	Assert Violation	Passed	Passed	Passed
SWC-111	Use of Deprecated Solidity Functions	Passed	Passed	Passed
SWC-112	Delegatecall to Untrusted Callee	Passed	Passed	Passed
SWC-113	DoS with Failed Call	Passed	Passed	Passed
SWC-114	Transaction Order Dependence	Passed	Passed	Passed
SWC-115	Authorization through tx.origin	Passed	Passed	Passed
SWC-116	Block values as a proxy for time	Passed	Passed	Passed
SWC-117	Signature Malleability	Passed	Passed	Passed
SWC-118	Incorrect Constructor Name	Passed	Passed	Passed
SWC-119	Shadowing State Variables	Passed	Passed	Passed
SWC-120	Weak Sources of Randomness from Chain Attributes	Passed	Passed	Passed

SWC-121	Missing Protection against Signature Replay Attacks	Passed	Passed	Passed
SWC-122	Lack of Proper Signature Verification	Passed	Passed	Passed
SWC-123	Requirement Violation	Passed	Passed	Passed
SWC-124	Write to Arbitrary Storage Location	Passed	Passed	Passed
SWC-125	Incorrect Inheritance Order	Passed	Passed	Passed
SWC-126	Insufficient Gas Griefing	Passed	Passed	Passed
SWC-127	Arbitrary Jump with Function Type Variable	Passed	Passed	Passed
SWC-128	DoS With Block Gas Limit	Passed	Passed	Passed
SWC-129	Typographical Error	low	Passed	Passed
SWC-130	Right-To-Left-Override control character (U+202E)	Passed	Passed	Passed
SWC-131	Presence of unused variables	Passed	Passed	Passed
SWC-132	Unexpected Ether balance	Passed	Passed	Passed
SWC-133	Hash Collisions With Multiple Variable Length Arguments	Passed	Passed	Passed
SWC-134	Message call with hardcoded gas amount	Passed	Passed	Passed
SWC-135	Code With No Effects	Passed	Passed	Passed
SWC-136	Unencrypted Private Data On-Chain	Passed	Passed	Passed

Detected High and Medium Severity Vulnerability Description.

Outdated Compiler Version.

Item: 1	Location:	Line 14	Severity:	 Low
---------	-----------	---------	-----------	---

Function	Using an outdated compiler version can be problematic especially if there are publicly disclosed bugs and issues that affect the current compiler version. The following outdated versions were detected: /aura.sol - 0.8.23
Remediation	It is recommended to use a recent version of the Solidity compiler that should not be the most recent version, and it should not be an outdated version as well. Using very old versions of Solidity prevents the benefits of bug fixes and newer security checks. Consider using the solidity version v0.8.27, which patches most solidity vulnerabilities.

⚠️ Incorrect Access Control (1 Item)

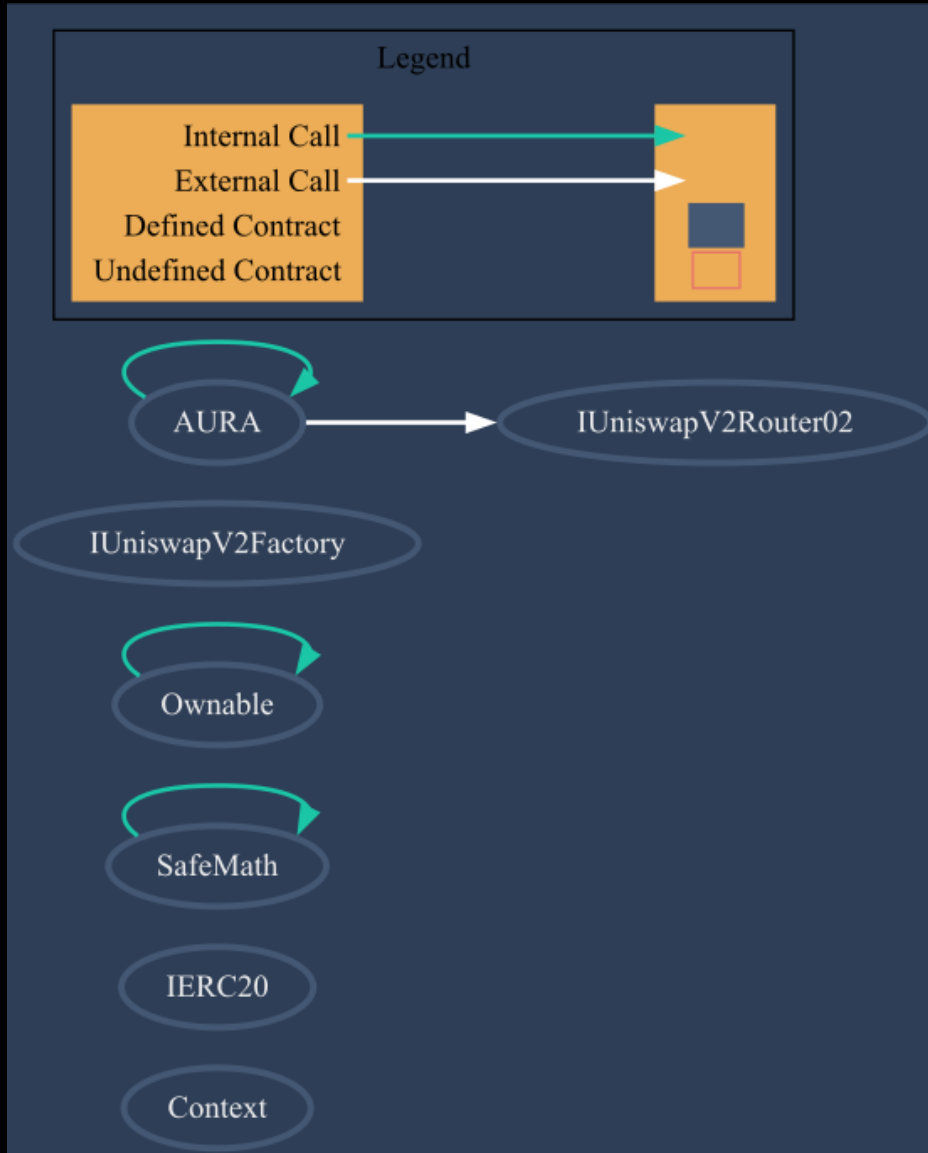
Item: 1	Location:	Line 342-352	Severity:	Medium
---------	-----------	--------------	-----------	--------

Function	<p>Access control plays an important role in segregation of privileges in smart contracts and other applications. If this is misconfigured or not properly validated on sensitive functions, it may lead to loss of funds, tokens and in some cases compromise of the smart contract.</p> <p>The contract AURA is importing an access control library @openzeppelin/contracts-upgradeable/access/OwnableUpgradeable.sol but the function manualswap is missing the modifier onlyOwner.</p>
Remedation	<ol style="list-style-type: none"> 1. Consider adding access control modifiers to the function to Ensure that initialization functions can only be called once and only by authorized entities. 2. Implement least-privilege roles using libraries like OpenZeppelin's Access Control. 3. Add proper access control modifiers to sensitive functions, such as onlyOwner or custom roles.

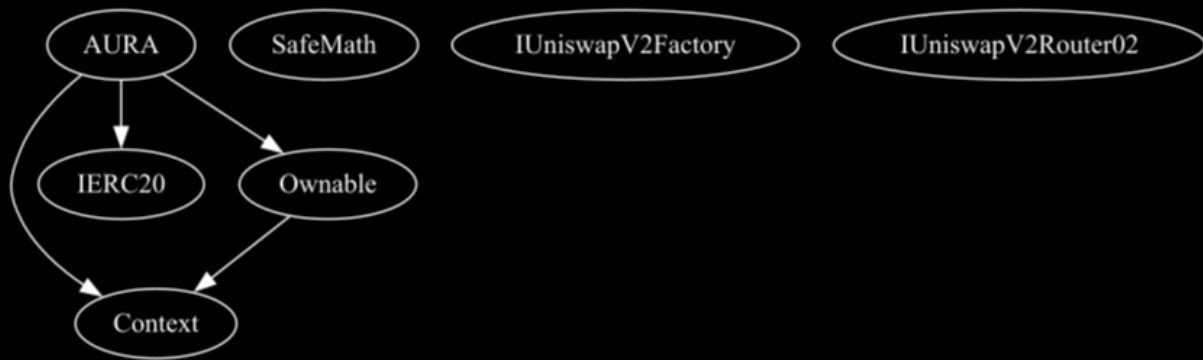
```

342     function manualswap() external {
343         require(_msgSender() == taxWallet);
344         uint256 tokenBalance = balanceOf(address(this));
345         if(tokenBalance > 0){
346             swapTokensForEth(tokenBalance);
347         }
348         uint256 ethBalance = address(this).balance;
349         if(ethBalance > 0){
350             sendETHToFee(ethBalance);
351         }
352     }
353 }
  
```





























Contract Interaction Graph





















































Inheritance Graph



Contract Functions

Contract	Type	Bases		
L	Function Name	Visibility	Mutability	Modifiers
Context	Implementation			
L	_msgSender	Internal 		
IERC20	Interface			
L	totalSupply	External 		NO 
L	balanceOf	External 		NO 
L	transfer	External 		NO 
L	allowance	External 		NO 
L	approve	External 		NO 
L	transferFrom	External 		NO 
SafeMath	Library			
L	add	Internal 		
L	sub	Internal 		
L	sub	Internal 		
L	mul	Internal 		
L	div	Internal 		
L	div	Internal 		
Ownable	Implementation	Context		
L		Public 		NO 
L	owner	Public 		NO 

L	renounceOwnership	Public 		onlyOwner
IUniswapV2Factory				
L	createPair	External 		NO 
IUniswapV2Router02				
L	swapExactTokensForETHSupportingFeeOnTransferTokens	External 		NO 
L	factory	External 		NO 
L	WETH	External 		NO 
L	addLiquidityETH	External 		NO 
AURA				
	Implementation	Context, IERC20, Ownable		
L		Public 		NO 
L	name	Public 		NO 
L	symbol	Public 		NO 
L	decimals	Public 		NO 
L	totalSupply	Public 		NO 
L	balanceOf	Public 		NO 
L	transfer	Public 		NO 
L	allowance	Public 		NO 
L	approve	Public 		NO 
L	transferFrom	Public 		NO 

L	_approve	Private 		
L	_transfer	Private 		
L	min	Private 		
L	swapTokensForEth	Private 		lockTheSwap
L	updateLimits	External 		onlyOwner
L	updateTransferTax	External 		onlyOwner
L	sendETHToFee	Private 		
L	addBots	Public 		onlyOwner
L	delBots	Public 		onlyOwner
L	isBot	Public 		NO 
L	Launch	External 		onlyOwner
L	updateFee	External 		NO 
L		External 		NO 
L	manualswap			



Function
can modify
state



Function
is payable

Audit Scope

Audit Method.

Our smart contract audit is an extensive methodical examination and analysis of the smart contract's code that is used to interact with the blockchain. Goal: discover errors, issues and security vulnerabilities in the code. Findings getting reported and improvements getting suggested.

Automatic and Manual Review

We are using automated tools to scan functions and weaknesses of the contract. Transfers, integer over-undeflow checks such as all CWE events.

Tools we use:

Visual Studio Code

CWE

SWC

Solidity Scan

SVD

In manual code review our auditor looking at source code and performing line by line examination. This method helps to clarify developer's coding decisions and business logic.

Skeleton Ecosystem

<https://skeletonecosystem.com>

<https://github.com/SkeletonEcosystem/Audits>

