

# Assure DeFi™

The Verification **Gold Standard**™



## Security Assessment **Alpha City Ai Token**

November 23, 2023

Audit Status: Fail

Audit Edition: Advance



ASSURE DEFI™  
THE VERIFICATION GOLD STANDARD

# Risk Analysis

## Classifications of Manual Risk Results

Classification	Description
🔴 Critical	Danger or Potential Problems.
🟠 High	Be Careful or Fail test.
🟡 Low	Pass, Not-Detected or Safe Item.
🔵 Informational	Function Detected

## Manual Code Review Risk Results

Contract Privilege	Description
🟢 Buy Tax	5
🟢 Sale Tax	5
🟢 Cannot Sale	Pass
🟢 Cannot Sale	Pass
🟠 Max Tax	20
🟠 Modify Tax	Detected
🟢 Fee Check	Pass
🟢 Is Honeypot?	Not detected
🟢 Trading Cooldown	Not Detected
🔴 Can Pause Trade?	Owner need to enableTrade.
🔴 Pause Transfer?	Detected
🟠 Max Tx?	Fail
🟠 Is Anti Whale?	Detected
🟠 Is Anti Bot?	Detected

Contract Privilege	Description
🔴 Is Blacklist?	Detected
🔴 Blacklist Check	Fail
🟠 is Whitelist?	Detected
🟢 Can Mint?	Pass
🟢 Is Proxy?	Not Detected
🟢 Can Take Ownership?	Not detected
🟢 Hidden Owner?	Not detected
🟡 Owner	0x
🟢 Self Destruct?	Not Detected
🟡 External Call?	Not detected
🟢 Other?	Not detected
🟢 Holders	2
🔴 Auditor Confidence	High Risk
🟡 KYC Present	No
🟡 KYC URL	

The following quick summary it's added to the project overview; however, there are more details about the audit and its results. Please read every detail.

# Project Overview

## Token Summary

Parameter	Result
Address	0x
Name	Alpha City Ai
Token Tracker	Alpha City Ai (AMETA)
Decimals	18
Supply	5,000,000,000
Platform	Ethereum
compiler	v0.8.10+commit.fc410830
Contract Name	AMETA
Optimization	Yes with 200 runs
LicenseType	MIT
Language	Solidity
Codebase	<a href="https://etherscan.io/address/0x#code">https://etherscan.io/address/0x#code</a>
Payment Tx	Corporate

## Main Contract Assessed Contract Name

Name	Contract	Live
Alpha City Ai	0x	Yes

## TestNet Contract Assessed Contract Name

Name	Contract	Live
Alpha City Ai	0x1787cE5f6eD4Ef08f3C9EB93D160B41E00bdB013	Yes

## Solidity Code Provided

Solid ID	File Sha-1	FileName
AMETA	78740c241f9a100cb4f249ec79bea23fc3922ff6	ameta.sol
AMETA		
AMETA		
AMETA		

# Smart Contract Vulnerability Checks

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 specific to smart contracts.

ID	Severity	Name	File	location
SWC-100	Pass	Function Default Visibility	ameta.sol	L: 0 C: 0
SWC-101	Pass	Integer Overflow and Underflow.	ameta.sol	L: 0 C: 0
SWC-102	Pass	Outdated Compiler Version file.	ameta.sol	L: 0 C: 0
SWC-103	Pass	A floating pragma is set.	ameta.sol	L: 0 C: 0
SWC-104	Pass	Unchecked Call Return Value.	ameta.sol	L: 0 C: 0
SWC-105	Pass	Unprotected Ether Withdrawal.	ameta.sol	L: 0 C: 0
SWC-106	Pass	Unprotected SELFDESTRUCT Instruction	ameta.sol	L: 0 C: 0
SWC-107	Pass	Read of persistent state following external call.	ameta.sol	L: 0 C: 0
SWC-108	Pass	State variable visibility is not set..	ameta.sol	L: 0 C: 0
SWC-109	Pass	Uninitialized Storage Pointer.	ameta.sol	L: 0 C: 0
SWC-110	Pass	Assert Violation.	ameta.sol	L: 0 C: 0
SWC-111	Pass	Use of Deprecated Solidity Functions.	ameta.sol	L: 0 C: 0
SWC-112	Pass	Delegate Call to Untrusted Callee.	ameta.sol	L: 0 C: 0
SWC-113	Pass	Multiple calls are executed in the same transaction.	ameta.sol	L: 0 C: 0

<b>ID</b>	<b>Severity</b>	<b>Name</b>	<b>File</b>	<b>location</b>
SWC-114	Pass	Transaction Order Dependence.	ameta.sol	L: 0 C: 0
SWC-115	Low	Authorization through tx.origin.	ameta.sol	L: 683 C: 57,L: 689 C: 53
SWC-116	Pass	A control flow decision is made based on The block.timestamp environment variable.	ameta.sol	L: 0 C: 0
SWC-117	Pass	Signature Malleability.	ameta.sol	L: 0 C: 0
SWC-118	Pass	Incorrect Constructor Name.	ameta.sol	L: 0 C: 0
SWC-119	Pass	Shadowing State Variables.	ameta.sol	L: 0 C: 0
SWC-120	Low	Potential use of block.number as source of randomness.	ameta.sol	L: 492 C: 29,L: 684 C: 32,L: 686 C: 32,L: 689 C: 66,L: 690 C: 59,L: 800 C: 15,L: 966 C: 29
SWC-121	Pass	Missing Protection against Signature Replay Attacks.	ameta.sol	L: 0 C: 0
SWC-122	Pass	Lack of Proper Signature Verification.	ameta.sol	L: 0 C: 0
SWC-123	Pass	Requirement Violation.	ameta.sol	L: 0 C: 0
SWC-124	Pass	Write to Arbitrary Storage Location.	ameta.sol	L: 0 C: 0
SWC-125	Pass	Incorrect Inheritance Order.	ameta.sol	L: 0 C: 0
SWC-126	Pass	Insufficient Gas Griefing.	ameta.sol	L: 0 C: 0
SWC-127	Pass	Arbitrary Jump with Function Type Variable.	ameta.sol	L: 0 C: 0
SWC-128	Pass	DoS With Block Gas Limit.	ameta.sol	L: 0 C: 0
SWC-129	Pass	Typographical Error.	ameta.sol	L: 0 C: 0

<b>ID</b>	<b>Severity</b>	<b>Name</b>	<b>File</b>	<b>location</b>
SWC-130	Pass	Right-To-Left-Override control character (U+202E).	ameta.sol	L: 0 C: 0
SWC-131	Pass	Presence of unused variables.	ameta.sol	L: 0 C: 0
SWC-132	Pass	Unexpected Ether balance.	ameta.sol	L: 0 C: 0
SWC-133	Pass	Hash Collisions with Multiple Variable Length Arguments.	ameta.sol	L: 0 C: 0
SWC-134	Pass	Message call with hardcoded gas amount.	ameta.sol	L: 0 C: 0
SWC-135	Pass	Code With No Effects (Irrelevant/Dead Code).	ameta.sol	L: 0 C: 0
SWC-136	Pass	Unencrypted Private Data On-Chain.	ameta.sol	L: 0 C: 0

We scan the contract for additional security issues using MYTHX and industry-standard security scanning tools.

# Smart Contract Vulnerability Details

## SWC-115 - Authorization through tx.origin

### CWE-477: Use of Obsolete Function

#### Description:

tx.origin is a global variable in Solidity which returns the address of the account that sent the transaction. Using the variable for authorization could make a contract vulnerable if an authorized account calls into a malicious contract. A call could be made to the vulnerable contract that passes the authorization check since tx.origin returns the original sender of the transaction which in this case is the authorized account.

#### Remediation:

tx.origin should not be used for authorization. Use msg.sender instead.

#### References:

Solidity Documentation - tx.origin

Ethereum Smart Contract Best Practices - Avoid using tx.origin

SigmaPrime - Visibility.

# **Smart Contract Vulnerability Details**

## **SWC-120 - Weak Sources of Randomness from Chain Attributes**

### **CWE-330: Use of Insufficiently Random Values**

#### **Description:**

Solidity allows for ambiguous naming of state variables when inheritance is used. Contract A with a variable x could inherit contract B that also has a state variable x defined. This would result in two separate versions of x, one of them being accessed from contract A and the other one from contract B. In more complex contract systems this condition could go unnoticed and subsequently lead to security issues.

Shadowing state variables can also occur within a single contract when there are multiple definitions on the contract and function level.

#### **Remediation:**

Using commitment scheme, e.g. RANDAO. Using external sources of randomness via oracles, e.g. Oraclize. Note that this approach requires trusting in oracle, thus it may be reasonable to use multiple oracles. Using Bitcoin block hashes, as they are more expensive to mine.

#### **References:**

How can I securely generate a random number in my smart contract?)

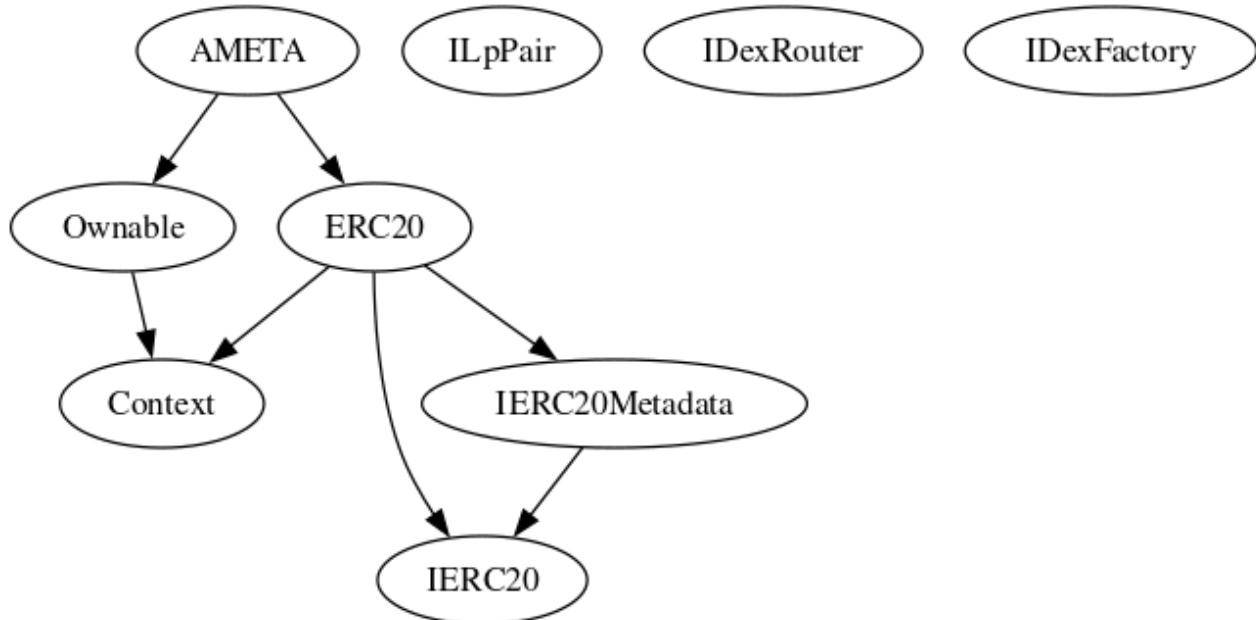
When can BLOCKHASH be safely used for a random number? When would it be unsafe?

The Run smart contract.

# Inheritance

The contract for Alpha City Ai has the following inheritance structure.

The Project has a Total Supply of 5,000,000,000



# Privileged Functions (onlyOwner)

Please Note if the contract is Renounced none of this functions can be executed.

Function Name	Parameters	Visibility
launch		external
disableMarkBotsForever		external
removeLimits		external
forceSwapBack		external
setTreasuryAddress		external
setOperationsAddress		external
withdrawStuckETH		external
transferForeignToken		external
excludeFromFees		external
updateSellFees		external
updateBuyFees		external
setAutomatedMarketMakerPair		external
updateSwapTokensAtAmount		external
updateMaxWalletAmount		external
updateMaxSellAmount		external

Function Name	Parameters	Visibility
setTaxFree		external
disableTransferDelay		external
emergencyUpdateRouter		external
removeBoughtEarly		external
markBoughtEarly		external
enableTrading		external

## AMETA-03 | Lack of Input Validation.

Category	Severity	Location	Status
Volatile Code	 Low	ameta.sol: L: 539 C: 14,L: 582 C: 14,L: 614 C: 14,L: 644 C: 14	 Detected

### Description

The given input is missing the check for the non-zero address.

The given input is missing the check for the missing required function.

### Remediation

We advise the client to add the check for the passed-in values to prevent unexpected errors as below:

```
...  
require(receiver != address(0), "Receiver is the zero address");  
...  
...  
require(value X limitation, "Your not able to do this function");  
...
```

We also recommend customer to review the following function that is missing a required validation. missing required function.

# AMETA-05 | Missing Event Emission.

Category	Severity	Location	Status
Volatile Code	 Low	ameta.sol: L: 947 C: 14,L: 632 C: 14,L: 620 C: 14,L: 539 C: 14,L: 510 C: 14,L: 515 C: 14,L: 570 C: 14	 Detected

## Description

Detected missing events for critical arithmetic parameters. There are functions that have no event emitted, so it is difficult to track off-chain changes. The linked code does not create an event for the transfer.

## Remediation

Emit an event for critical parameter changes. It is recommended emitting events for the sensitive functions that are controlled by centralization roles.

## **AMETA-11 | identifiedBots can be used as blacklist within code and tax up to 80%.**

Category	Severity	Location	Status
Optimization	<span style="color: red;">●</span> Critical	ameta.sol: L: 501 C: 14	<span style="color: blue;">[ ]</span> Detected

### **Description**

identifiedBots can be mis used for blacklist and get holders tax up to 80%, this should be carefully used otherwise can negatively impact the project.

### **Remediation**

Recommended to remove markBoughtEarly function and only leave the remove just in case the bot detect the incorrect holder.

### **Project Action**

## AMETA-13 | Extra Gas Cost For User.

Category	Severity	Location	Status
Logical Issue	 Informational	ameta.sol: L: 731 C: 14	 Detected

### Description

The user may trigger a tax distribution during the transfer process, which will cost a lot of gas and it is unfair to let a single user bear it.

### Remediation

We advise the client to make the owner responsible for the gas costs of the tax distribution.

### Project Action

## **AMETA-18 | Stop Transactions by using Enable Trade.**

<b>Category</b>	<b>Severity</b>	<b>Location</b>	<b>Status</b>
Logical Issue	 Critical	ameta.sol: L: 484 C:14	 Detected

### **Description**

Enable Trade is present on the following contract and when combined with Exclude from fees it can be considered a whitelist process, this will allow anyone to trade before others and can represent and issue for the holders.

### **Remediation**

We recommend the project owner to carefully review this function and avoid problems when performing both actions.

### **Project Action**

# Technical Findings Summary

## Classification of Risk

Severity	Description
🔴 Critical	Risks are those that impact the safe functioning of a platform and must be addressed before launch. Users should not invest in any project with outstanding critical risks.
🟠 High	Risks can include centralization issues and logical errors. Under specific circumstances, these major risks can lead to loss of funds and/or control of the project.
🟡 Medium	Risks may not pose a direct risk to users' funds, but they can affect the overall functioning of a platform
⚡ Low	Risks can be any of the above but on a smaller scale. They generally do not compromise the overall integrity of the Project, but they may be less efficient than other solutions.
ℹ️ Informational	Errors are often recommended to improve the code's style or certain operations to fall within industry best practices. They usually do not affect the overall functioning of the code.

## Findings

Severity	Found	Pending	Resolved
🔴 Critical	2	2	1
🟠 High	0	0	1
🟡 Medium	0	0	0
⚡ Low	2	2	0
ℹ️ Informational	1	1	0
Total	5	5	2

# Social Media Checks

Social Media	URL	Result
Twitter	<a href="https://x.com/alphametaeco?s=21&amp;t=dxV-X5ryK1awxv11lInjPg">https://x.com/alphametaeco?s=21&amp;t=dxV-X5ryK1awxv11lInjPg</a>	Pass
Other		Fail
Website	<a href="https://www.alphacitymeta.com">https://www.alphacitymeta.com</a>	Pass
Telegram	<a href="https://t.me/alphacityai">https://t.me/alphacityai</a>	Pass

We recommend to have 3 or more social media sources including a completed working websites.

**Social Media Information Notes:**

**Auditor Notes:** undefined

**Project Owner Notes:**



# **Assessment Results**

## **Score Results**

Review	Score
Overall Score	66/100
Auditor Score	65/100
Review by Section	Score
Manual Scan Score	10
SWC Scan Score	33
Advance Check Score	23

The Following Score System Has been Added to this page to help understand the value of the audit, the maximum score is 100, however to attain that value the project must pass and provide all the data needed for the assessment. Our Passing Score has been changed to 84 Points for a higher standard, if a project does not attain 85% is an automatic failure. Read our notes and final assessment below.

## **Audit Fail**



# **Assessment Results**

## **Important Notes:**

- No vulnerabilities or exploits were detected at the time of the audit.
- Several items need improvements in this code.
- The developer addressed some key items in this revision.
- The code has an aggressive anti-bot that can be used as a blacklist.

**Auditor Score =65**  
**Audit Fail**



# Appendix

## Finding Categories

### Centralization / Privilege

Centralization / Privilege findings refer to either feature logic or implementation of components that act against the nature of decentralization, such as explicit ownership or specialized access roles in combination with a mechanism to relocate funds.

### Gas Optimization

Gas Optimization findings do not affect the functionality of the code but generate different, more optimal EVM opcodes resulting in a reduction on the total gas cost of a transaction.

### Logical Issue

Logical Issue findings detail a fault in the logic of the linked code, such as an incorrect notion on how `block.timestamp` works.

### Control Flow

Control Flow findings concern the access control imposed on functions, such as owner-only functions being invoke-able by anyone under certain circumstances.

### Volatile Code

Volatile Code findings refer to segments of code that behave unexpectedly on certain edge cases that may result in a vulnerability.

### Coding Style

Coding Style findings usually do not affect the generated byte-code but rather comment on how to make the codebase more legible and, as a result, easily maintainable.

### Inconsistency

Inconsistency findings refer to functions that should seemingly behave similarly yet contain different code, such as a constructor assignment imposing different require statements on the input variables than a setter function.

### Coding Best Practices

ERC 20 Coding Standards are a set of rules that each developer should follow to ensure the code meets a set of criteria and is readable by all the developers.

# Disclaimer

Assure Defi has conducted an independent security assessment to verify the integrity of and highlight any vulnerabilities or errors, intentional or unintentional, that may be present in the reviewed code for the scope of this assessment. This report does not constitute agreement, acceptance, or advocacy for the Project, and users relying on this report should not consider this as having any merit for financial advice in any shape, form, or nature. The contracts audited do not account for any economic developments that the Project in question may pursue, and the veracity of the findings thus presented in this report relate solely to the proficiency, competence, aptitude, and discretion of our independent auditors, who make no guarantees nor assurance that the contracts are entirely free of exploits, bugs, vulnerabilities or depreciation of technologies.

All information provided in this report does not constitute financial or investment advice, nor should it be used to signal that any persons reading this report should invest their funds without sufficient individual due diligence, regardless of the findings presented. Information is provided ‘as is, and Assure Defi is under no covenant to audited completeness, accuracy, or solidity of the contracts. In no event will Assure Defi or its partners, employees, agents, or parties related to the provision of this audit report be liable to any parties for, or lack thereof, decisions or actions with regards to the information provided in this audit report.

The assessment services provided by Assure Defi are subject to dependencies and are under continuing development. You agree that your access or use, including but not limited to any services, reports, and materials, will be at your sole risk on an as-is, where-is, and as-available basis. Cryptographic tokens are emergent technologies with high levels of technical risk and uncertainty. The assessment reports could include false positives, negatives, and unpredictable results. The services may access, and depend upon, multiple layers of third parties.

