



Security Assessment

ARBITRUM EXCHANGE

Vital Block Verified on March 14th, 2023

 @Vital-Block

 @VB_Audit

 info@vitalblock.org



 www.vitalblock.org



PREPARED FOR:
ARBITRUM
EXCHANGE



INTRODUCTION

Auditing Firm	 VITAL BLOCK SECURITY
Client Firm	 ABITRUM EXCHANGE
Methodology	Automated Analysis, Manual Code Review
Language	Solidity
Contract	<p>PRESALE (ASAP) TOKEN: 0x94e0E99759753D4aD17e508cf7Ee25d2eA002486</p> <p>factory: factory: 0x1C6E968f2E6c9DEC61DB874E28589fd5CE3E1f2c</p> <p>Multi-Sig: 0xE8FFE751deA181025a9ACf3D6Bde8cdA5380F53F</p> <p>masterchef: 0x5693a243b0968e4218faF590bdA4607E65Ff499d</p> <p>Timelock: 0x6D6dA6f5D018341899187eD0A9F38790bd92aF3d</p> <p>Presale: 0x751892588082B2a217db19769a438570e7aEBd4E</p>
Blockchain	ARBITRUM
Centralization	Active ownership
Website	https://arbidex.fi
Discord	https://discord.com/MjDrbBc7GC
Twitter	https://twitter.com/Arbidex_fi
GitHub	https://github.com/fractality/arbidex
Prelim Report Date	February 8, 2023
Final Report Date	March 14, 2023



Verify the authenticity of this report on our GitHub Repo: <https://www.github.com/vital-block>

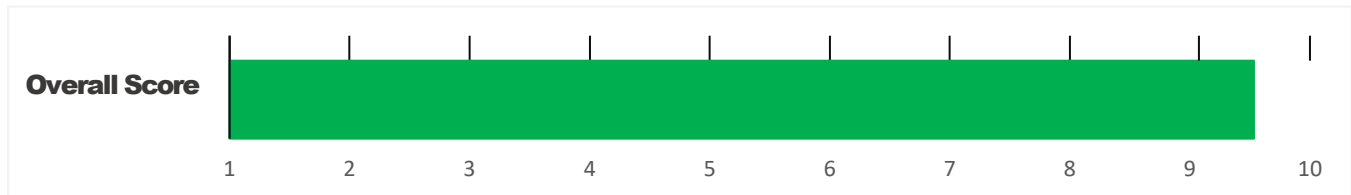


EXECUTIVE SUMMARY

Vital Block Security has performed the automated and manual analysis of the Sol code. The code was reviewed for common contract vulnerabilities and centralized exploits. Here's a quick audit summary:

Status	Critical ! 🔴	Major " 🟡	Medium # 🟡	Minor \$ 🟢	Unknown % 🟤
Open	0	0	0	2	0
Acknowledged	0	0	2	6	0
Resolved	0	0	0	0	0
Noteworthy onlyOwner Privileges	Set Taxes and Ratios, Airdrop, Set Protection Settings, Set Reward Properties, Set Reflector Settings, Set Swap Settings, Set Pair and Router				

ARBITRUM EXCHANGE Smart contract has achieved the following score: **95.0**



i Please note that smart contracts deployed on blockchains aren't resistant to exploits, vulnerabilities and/or hacks. Blockchain and cryptography assets utilize new and emerging technologies. These technologies present a high level of ongoing risks. For a detailed understanding of risk severity, source code vulnerability, and audit limitations, kindly review the audit report thoroughly.

i Please note that centralization privileges regardless of their inherited risk status - constitute an elevated impact on smart contract safety and security.



TABLE OF CONTENTS

TABLE OF CONTENTS	4
SCOPE OF WORK	5
AUDIT METHODOLOGY	6
RISK CATEGORIES	8
CENTRALIZED PRIVILEGES	9
AUTOMATED ANALYSIS	10
INHERITANCE GRAPH	15
MANUAL REVIEW	16
DISCLAIMERS	27
ABOUT VITALBLOCK	30



SCOPE OF WORK

Vital Block was consulted by **ABITRUM EXCHANGE** to conduct the smart contract audit of its. Sol source code. The audit scope of work is strictly limited to mentioned .SOL file only:

- **ARBDEX.Sol**

 **External contracts and/or interfaces dependencies are not checked due to being out of scope.**

Verify audited contract's contract address and deployed link below:

Public Contract Link	
<u>PRESALE: 0x751892588082B2a217db19769a438570e7aEBd4E</u>	
<u>ARX: 0x94e0E99759753D4aD17e508cf7Ee25d2eA002486</u>	
Contract Name	ARBITRUM EXCHANGE
Token Symbol	ARX



AUDIT METHODOLOGY

Smart contract audits are conducted using a set of standards and procedures. Mutual collaboration is essential to performing an effective smart contract audit. Here's a brief overview of Vital Block auditing process and methodology:

CONNECT

- The onboarding team gathers source codes, and specifications to make sure we understand the size, and scope of the smart contract audit.

AUDIT

- Automated analysis is performed to identify common contract vulnerabilities. We may use the following third-party frameworks and dependencies to perform the automated analysis:
 - Remix IDE Developer Tool
 - Open Zeppelin Code Analyzer
 - SWC Vulnerabilities Registry
 - DEX Dependencies, e.g., Pancakeswap, Uniswap
- Simulations are performed to identify centralized exploits causing contract and/or trade locks.
- A manual line-by-line analysis is performed to identify contract issues and centralized privileges.

We may inspect below mentioned common contract vulnerabilities, and centralized exploits:

Centralized Exploits	<ul style="list-style-type: none">○ Token Supply Manipulation○ Access Control and Authorization○ Assets Manipulation○ Ownership Control○ Liquidity Access○ Stop and Pause Trading○ Ownable Library Verification
----------------------	---



Common Contract Vulnerabilities

- **Integer Overflow**
- **Lack of Arbitrary limits**
- **Incorrect Inheritance Order**
- **Typographical Errors**
- **Requirement Violation**
- **Gas Optimization**
- **Coding Style Violations**
- **Re-entrancy**
- **Third-Party Dependencies**
- **Potential Sandwich Attacks**
- **Irrelevant Codes**
- **Divide before multiply**
- **Conformance to Solidity Naming Guides**
- **Compiler Specific Warnings**
- **Language Specific Warnings**

REPORT

- **The auditing team provides a preliminary report specifying all the checks which have been performed and the findings thereof.**
- **The client's development team reviews the report and makes amendments to the codes.**
- **The auditing team provides the final comprehensive report with open and unresolved issues.**

PUBLISH






- **The client may use the audit report internally or disclose it publicly.**

 **It is important to note that there is no pass or fail in the audit, it is recommended to view the audit as an unbiased assessment of the safety of solidity codes.**



RISK CATEGORIES

Smart contracts are generally designed to hold, approve, and transfer tokens. This makes them very tempting attack targets. A successful external attack may allow the external attacker to directly exploit. A successful centralization-related exploit may allow the privileged role to directly exploit. All risks which are identified in the audit report are categorized here for the reader to review:

Risk Type	Definition
Critical ! 	These risks could be exploited easily and can lead to asset loss, data loss, asset, or data manipulation. They should be fixed right away.
Major " 	These risks are hard to exploit but very important to fix, they carry an elevated risk of smart contract manipulation, which can lead to high-risk severity.
Medium # 	These risks should be fixed, as they carry an inherent risk of future exploits, and hacks which may or may not impact the smart contract execution. Low-risk re-entrancy-related vulnerabilities should be fixed to deter exploits.
Minor \$ 	These risks do not pose a considerable risk to the contract or those who interact with it. They are code-style violations and deviations from standard practices. They should be highlighted and fixed nonetheless.
Unknown % 	These risks pose uncertain severity to the contract or those who interact with it. They should be fixed immediately to mitigate the risk uncertainty.

All statuses which are identified in the audit report are categorized here for the reader to review:

Status Type	Definition
Open	Risks are open.
Acknowledged	Risks are acknowledged, but not fixed.
Resolved	Risks are acknowledged and fixed.



CENTRALIZED PRIVILEGES


Centralization risk is the most common cause of cryptography asset loss. When a smart contract has a privileged role, the risk related to centralization is elevated.

There are some well-intended reasons have privileged roles, such as:

- **Privileged roles can be granted the power to `pause()` the contract in case of an external attack.**
- **Privileged roles can use functions like, `include()`, and `exclude()` to add or remove wallets from fees, swap checks, and transaction limits. This is useful to run a presale and to list on an exchange.**

Authorizing privileged roles to externally-owned-account (EOA) is dangerous. Lately, centralization-related losses are increasing in frequency and magnitude.

- **The client can lower centralization-related risks by implementing below mentioned practices:**
- **Privileged role's private key must be carefully secured to avoid any potential hack.**
- **Privileged role should be shared by multi-signature (multi-sig) wallets.**
- **Authorized privilege can be locked in a contract, user voting, or community DAO can be introduced to unlock the privilege.**
- **Renouncing the contract ownership, and privileged roles.**
- **Remove functions with elevated centralization risk.**

 **Understand the project's initial asset distribution. Assets in the liquidity pair should be locked. Assets outside the liquidity pair should be locked with a release schedule.**








AUDIT SCOPE

ARBITRUM EXCHANGE

ID	Repo	Comment	File	SHM321 Checksum
ABU	contracts/base/core	cC512486	ARXPresale.sol	85f15802c6be0fd50f8632d8433ccc9db6f4b39f9e566d1fa78de54b84bdd35
ABH	contracts/base/core	cC512486	ARXPool.sol	8oipkjjk96be0fd50f8632d8433ccc9db6f4b39f9e566d1yhhg8765fffcuiuybb
ABV	contracts/base/core	cC512486	ARXToken.sol	3666778uj908766362fvyga98jdkl88648yhfbqt37409owehbgwhuyyyg223738
ABT	contracts/base/core	cC512486	ArbDexChef.sol	98uuyriy399787390uhbiiuhghhdg7guu30oi7799u9359ydfgdgygeigi3ioueyy78
ABI	contracts/base/token	cC512486	ArbDexFactory.sol	4566efgywqutfeuh87872t15378837983639293763hhegetgjfwjk89336668862
ABP	contracts/base/token	cC512486	SmartChefInitializable.sol	546363ttebnve88329973mvvdsggc478153ytdgfdxy792635fgdjgi1900990908
ABN	contracts/base/vote	cC512486	SmartChefInitializable.solStr Voter.sol	835656990327hudbinnjnr6729dc4jld0993ytyy3vq63235727879889073
ABY	contracts/base/vote	cC512486	Timelock.sol	cc089692343d1cc36eaf196046d7a528d153abd55ba20e82f1d57c22fcd92675
ABO	contracts/base/vote	cC512486	Timelock:.sol	8448b3af42497f5f74e53424ee3e6c551f51356945108d22a893d608a7990542
ABX	contracts/base/reward	cC512486	Multi-Sig	5c86aa1dd3889db5fcd17a80214b226fc784f268ab9db82df97c1d2459467831
ABF	contracts/base/reward	cC512486	MasterChefBribeFactory.sol	d7twtry997600965gd445667788954rewadfhjnbvct5566774332234567fdvffaq
ABW	contracts/base/presale	cC512486	Presale.sol	3d408b8f2cc56f9699a402b5151de90671de089c3007afc9e4fc867c04152e7c
ABQ	contracts/base/	cC512486	presale ASAP Token.sol	9d751621c3501102e4b50005ca3314ec6e04e6ff8bbb30852d1c7edfff3f8cef
ABL	contracts/base/reward	cC512480	MultiRewardsPoolBase.sol	455687gfasadjknppiiuhhg774580vgfxrki9876dhgvb990lkjhde444566788



AUTOMATED ANALYSIS

Symbol	Definition
	Function modifies state
	Function is payable
	Function is internal
	Function is private
	Function is important

```

**ARBDEX** | Interface | |||
| L | totalSupply | External | ! | NO |
| L | decimals | External | ! | NO |
| L | symbol | External | ! | NO |
| L | name | External | ! | NO |
| L | getOwner | External | NO |
| L | balanceOf | External | ! | NO |
| L | transfer | External | " ! | NO |
| L | allowance | External | ! | NO |
| L | approve | External | " ! | NO |
| L | transferFrom | External | " | NO |
|||||
**IFactoryV2** | Interface | |||
| L | getPair | External | NO | |
| L | createPair | External | " | NO |
|||||
**IV2Pair** | Interface | |||
| L | factory | External | NO | |
| L | getReserves | External | NO |
| L | sync | External | " | NO |

```



```
|||||
```

```

IRouter01 Interface
  L factory External | NO
  L ETH External | NO
  L addLiquidityETH External | # NO
  L addLiquidity External | " NO
  L swapExactAPForTokens External | # NO
  L getAmountsOut External | NO
  L getAmountsIn External | NO

```

```
|||||
```

```

IRouter02 Interface IRouter01
  L swapExactTokensForETHSupportingFeeOnTransferTokens External | " NO
  L swapExactETHForTokensSupportingFeeOnTransferTokens External | # NO
  L swapExactTokensForTokensSupportingFeeOnTransferTokens External | " ! NO
  L swapExactTokensForTokens External | " NO

```

```
|||||
```

```

Protections Interface
  L checkUser External | " ! NO
  L setLaunch External | " NO
  L setLpPair External | " NO
  L ARX External | " NO
  L removeSniper External | " NO

```

```
|||||
```

```

Cashier Interface
  L setRewardsProperties External | " NO
  L tally External | " NO
  L load External | # NO
  L cashout External | " NO
  L giveMeWelfarePlease External | " NO
  L getTotalDistributed External | NO
  L getUserInfo External | NO
  L getUserRealizedRewards External | NO

```

```

| L | getPendingRewards | External | | | NO |
| L | initialize | External | | " | NO |
| L | getCurrentReward | External | | | NO |
|||||
| **SOL** | Implementation | SafeMath | | |
| L | <Constructor> | Public | | # | NO |
| L | transferOwner | External | | " | onlyOwner |
| L | renounceOwnership | External | | " | NO |
| L | setOperator | Public | | " | NO |
| L | renounceOriginalDeployer | External | | " | NO |
| L | <Receive Ether> | External | | # | NO |
| L | totalSupply | External | | | NO |
| L | decimals | External | | | NO |
| L | symbol | External | | | NO |
| L | name | External | | | NO |
| L | getOwner | External | | ! | NO |
| L | balanceOf | Public | | ! | NO |
| L | allowance | External | | ! | NO |
| L | approve | External | | " ! | NO |
| L | _approve | Internal | $ | " | |
| L | approveContractContingency | Public | | " ! | onlyOwner |
| L | transfer | External | | " | NO |
| L | transferFrom | External | | " | NO |
| L | setNewRouter | External | | " | onlyOwner |
| L | setLpPair | External | | " | onlyOwner |
| L | setInitializers | External | | " | onlyOwner |
| L | isExcludedFromFees | External | | | NO |
| L | isExcludedFromDividends | External | | | NO |
| L | isExcludedFromProtection | External | | | NO |
| L | setDividendExcluded | Public | | " | onlyOwner |
| L | setExcludedFromFees | Public | | " | onlyOwner |

```

ABU-01 POSSIBLE OVERFLOW

Category	Severity ●	Location	Status
StatusMathematical Operations	Minor	contracts/base/ARXPresale.sol	Acknowledged

Description

In `updateForTaker`, the following equation is used inside an unchecked block






```
function functionCall(address target, bytes memory data, string memory errorMessage) internal returns (bytes memory) {  
    return function CallWithValue(target, data, 0, errorMessage);  
}
```

Where `parameters.amountOutUsed` is a `memory` and `override In` is a `memory` As these two are multiplied together in an unchecked block, they may overflow.

Recommendation

We recommend either checking for overflow in this case, or ensuring that the `Pairs In` is close enough it will never cause an overflow

OPTIMIZATIONS | ARBITRUM EXCHANGE

ID	Title	Category	Status
STV	Logarithm Refinement Optimization	Gas Optimization	Acknowledged 
SOP	Checks Can Be Performed Earlier	Gas Optimization	Acknowledged 
SDP	Unnecessary Use Of SafeMath	Gas Optimization	Acknowledged 
SWY	Struct Optimization	Gas Optimization	Acknowledged 
SGT	Unused State Variable	Gas Optimization	Acknowledged 

General Detectors



Floating Pragma

This contract may not function as expected due to inconsistent solidity compiler versions being specified



Attention
Required



Low Level Calls

This contract uses low-level calls, which may be unsafe.



Attention
Required



Numeric Notation Best Practices

The numeric notation used in this contract is unconventional, possibly worsening the reading/debugging experience



Attention
Required

- | | |
|--|--|
| ✓ No compiler version inconsistencies found | ✓ No tautologies or contradictions found |
| ✓ No unchecked call responses found | ✓ No faulty true/false values found |
| ✓ No vulnerable self-destruct functions found | ✓ No inaccurate divisions found |
| ✓ No assertion vulnerabilities found | ✓ No redundant constructor calls found |
| ✓ No old solidity code found | ✓ No vulnerable transfers found |
| ✓ No external delegated calls found | ✓ No vulnerable return values found |
| ✓ No external call dependency found | ✓ No uninitialized local variables found |
| ✓ No vulnerable authentication calls found | ✓ No default function responses found |
| ✓ No invalid character typos found | ✓ No missing arithmetic events found |
| ✓ No RTL characters found | ✓ No missing access control events found |
| ✓ No dead code found | ✓ No redundant true/false comparisons found |
| ✓ No risky data allocation found | ✓ No state variables vulnerable through function calls found |
| ✓ No uninitialized state variables found | ✓ No buggy low-level calls found |
| ✓ No uninitialized storage variables found | ✓ No expensive loops found |
| ✓ No vulnerable initialization functions found | ✓ No bad numeric notation practices found |
| ✓ No risky data handling found | ✓ No missing constant declarations found |
| ✓ No number accuracy bug found | ✓ No missing external function declarations found |
| ✓ No suspicious Masterchef approvals found | ✓ No vulnerable payable functions found |
| ✓ Withdrawals cannot be paused | ✓ No vulnerable message values found |



General Detectors | Master Chef



The score provides an indication of how safe the contract is based on our analysis. We take into account both the amount and the severity of detected issues. Please use this as a guideline only and always DYOR.

masterchef: 0x5693a243b0968e4218faF590bdA4607E65Ff499d



State Variables Should be Declared Constant

Some state variables in this contract should be declared as constant.



Attention
Required



Public Functions Should be Declared External

Some functions in this contract should be declared as external in order to save gas.



Attention
Required

- | | |
|--|---|
| ✓ No vulnerable withdrawal functions found | ✓ No unchecked call responses found |
| ✓ No reentrancy risk found | ✓ No vulnerable self-destruct functions found |
| ✓ No locks detected | ✓ No assertion vulnerabilities found |
| ✓ Verified source code found | ✓ No old solidity code found |
| ✓ No migration risk found | ✓ No external delegated calls found |
| ✓ Contract cannot be upgraded | ✓ No vulnerable authentication calls found |
| ✓ No privileged withdrawals are allowed | ✓ No invalid character typos found |
| ✓ No deposit fee found | ✓ No RTL characters found |
| ✓ No withdrawal fee found | ✓ No dead code found |
| ✓ No ERC20 approval vulnerability found | ✓ No risky data allocation found |
| ✓ Contract owner cannot abuse ERC20 approvals | ✓ No uninitialized storage variables found |
| ✓ No risky data allocation found | ✓ No risky data handling found |
| ✓ No blocking loops found | ✓ No number accuracy bug found |
| ✓ Wallets cannot be blacklisted from any specific contract functionality | ✓ No out-of-range number vulnerability found |
| ✓ No vulnerable initialization functions found | ✓ No map data deletion vulnerabilities found |
| ✓ Withdrawals cannot be paused | ✓ No tautologies or contradictions found |
| ✓ No suspicious Masterchef approvals found | ✓ No faulty true/false values found |
| ✓ No approval restrictions found | ✓ No redundant constructor calls found |
| ✓ Withdrawals cannot be paused | ✓ No vulnerable return values found |
| ✓ No uninitialized local variables found | |



Vulnerability Scan

REENTRANCY

✓ No reentrancy risk found

Severity Major

Confidence Parameter Certain

✗ **Mintable**: More amount of this token can be minted by a private wallet or contract. (This is Essentially normal for most contracts)

```
function _functionCallWithValue(
    address target,
    bytes memory data,
    uint256 weiValue,
    string memory errorMessage
) private returns (bytes memory) {
    require(isContract(target), 'Address: call to non-contract');

    // solhint-disable-next-line avoid-low-level-calls
    (bool success, bytes memory returndata) = target.call{value:
weiValue}(data);
    if (success) {
        return returndata;
    } else {
        // Look for revert reason and bubble it up if present
        if (returndata.length > 0) {
            // The easiest way to bubble the revert reason is using memory via
assembly

            // solhint-disable-next-line no-inline-assembly
            assembly {
                let returndata_size := mload(returndata)
                revert(add(32, returndata), returndata_size)
            }
        } else {

```

Vulnerability Description

Scanning Line:

Repository:

<https://github.com/kevinjurden/ArbDex/tree/main>

All Audited Files

Token.sol
Factory.sol
Multi-sig.sol
masterchef.sol
Timelock.sol
Presale.sol

Contract Creator

0x9aa54193aa3b23262e737c853e93e6b364a987a4

Creator Txn Hash

0xd7e9e03c750c4d2c8fb8655c1f657c1747904c8dd524999b71e8ddc4632bb121

Contracts:

Contract

PRESALE (ASAP)TOKEN: 0x94e0E99759753D4aD17e508cf7Ee25d2eA002486
factory: 0x1C6E968f2E6c9DEC61DB874E28589fd5CE3E1f2c
Multi-Sig: 0xE8FFE751deA181025a9ACf3D6Bde8cdA5380F53F
masterchef: 0x5693a243b0968e4218faF590bdA4607E65Ff499d
Timelock: 0x6D6dA6f5D018341899187eD0A9F38790bd92aF3d
Presale: 0x751892588082B2a217db19769a438570e7aEBd4E



Vulnerability Run check

ArbDex Token / ARX

08/03/2023 04:19 AM UTC+8

Contract Info

Total supply 0
Transaction Tax Buy 0.00% / Sell 0.00%

Risk Analysis

✔ Contract source code verified

This token contract is open source. You can check the contract code for details. Unsourced token contracts are likely to have malicious functions to defraud their users of their assets.

✔ No Proxy

There is no proxy in the contract. The proxy contract means contract owner can modify the function of the token and possibly effect the price.

⚠ Mint function

The contract may contain additional issuance functions, which could maybe generate a large number of tokens, resulting in significant fluctuations in token prices. It is recommended to confirm with the project team whether it complies with the token issuance instructions.

✔ No function to retrieve ownership

If this function exists, it is possible for the project owner to regain ownership even after relinquishing it.

✔ Owner cant change balance

The contract owner does not have the authority to modify the balance of tokens at other addresses.

✔ No whitelist function

Whitelist function found

Honeypot Risk

✔ This does not appear to be a honeypot

We are not aware of any code that prevents the sale of tokens.

✔ No trading cooldown

The token contract has no trading cooldown function.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.

✔ No Anti Whale

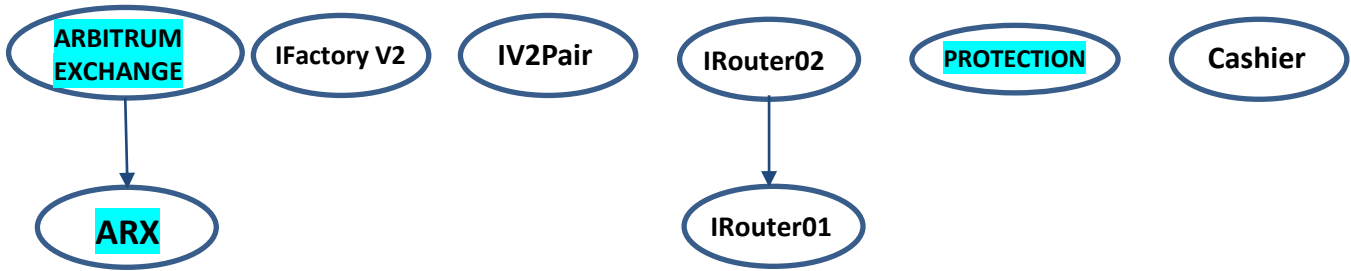
There is no limit to the number of token transactions. The number of scam token transactions may be limited (honeypot risk).

✔ No blacklist function

No blacklist function is included.



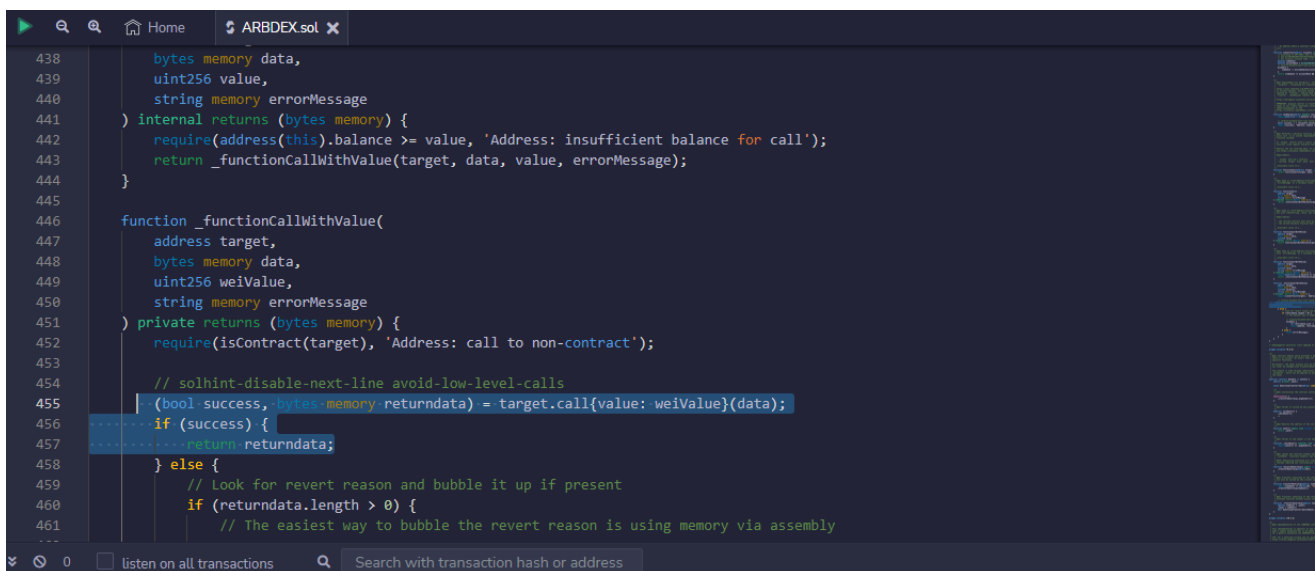
INHERITANCE GRAPH



Identifier	Definition	Severity
CEN-12	Centralization privileges of ARBITRUM EXCHANGE	Medium # 

Vulnerability 0 : No important security issue detected.

Threat level: Low



```

438     bytes memory data,
439     uint256 value,
440     string memory errorMessage
441 ) internal returns (bytes memory) {
442     require(address(this).balance >= value, 'Address: insufficient balance for call');
443     return _functionCallWithValue(target, data, value, errorMessage);
444 }
445
446 function _functionCallWithValue(
447     address target,
448     bytes memory data,
449     uint256 weiValue,
450     string memory errorMessage
451 ) private returns (bytes memory) {
452     require(isContract(target), 'Address: call to non-contract');
453
454     // solhint-disable-next-line avoid-low-level-calls
455     (bool success, bytes memory returndata) = target.call{value: weiValue}(data);
456     if (success) {
457         return returndata;
458     } else {
459         // Look for revert reason and bubble it up if present
460         if (returndata.length > 0) {
461             // The easiest way to bubble the revert reason is using memory via assembly
462         }
463     }
464 }
  
```

MANUAL REVIEW

ARBITRUM EXCHANGE: ARBDEX IS THE MOST SECURE COMMUNITY-DRIVEN REWARDING DEX ON ARBITRUM NETWORK.

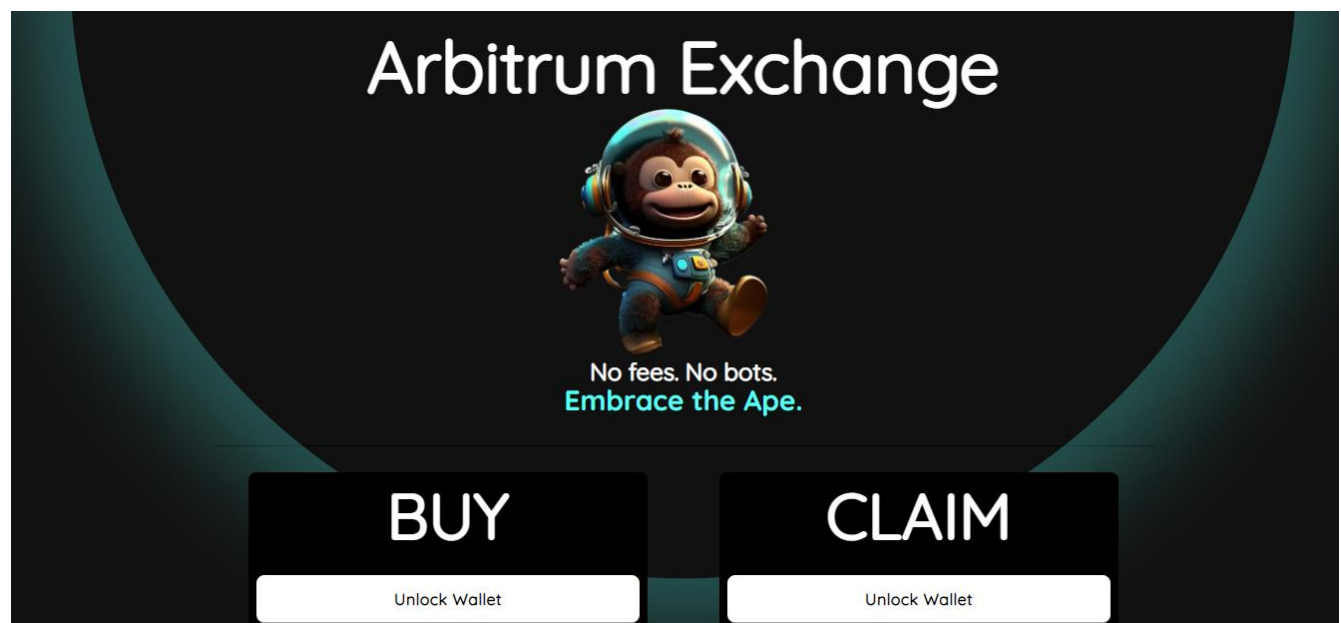
TOKEN NAME: ARBDEX

Ticker: ARX

Chain/Standard: ARBITRUM BLOCKCHAIN



The ARBITRUM EXCHANGE Platform Is Launched On Arbitrum





ISSUES CHECKING STATUS

Issue Description

Checking Status

1.	Compiler errors.	PASSED
2.	Race Conditions and reentrancy. Cross-Function Race Conditions.	PASSED
3.	Possible Delay In Data Delivery.	PASSED
4.	Oracle calls.	PASSED
5.	Front Running.	PASSED
6.	Sol Dependency.	PASSED
7.	Integer Overflow And Underflow.	PASSED
8.	DoS with Revert.	PASSED
9.	Dos With Block Gas Limit.	PASSED
10.	Methods execution permissions.	PASSED
11.	Economy Model of the contract.	PASSED
12.	The Impact Of Exchange Rate On the solidity Logic.	PASSED
13.	Private use data leaks.	PASSED
14.	Malicious Event log.	PASSED
15.	Scoping and Declarations.	PASSED
16.	Uninitialized storage pointers.	PASSED
17.	Arithmetic accuracy.	PASSED
18.	Design Logic.	PASSED
19.	Cross-Function race Conditions	PASSED
20.	Save Upon solidity contract Implementation and Usage.	PASSED
21.	Fallback Function Security	PASSED



AUDIT RESULT

PASSED

SMART CONTRACT AUDIT OF ARBDEX

Identifier	Definition	Severity
CEN-02	Initial asset distribution	Minor 

All of the initially minted assets are sent to the contract deployer when deploying the contract. This is Normal for most deployer and/or contract owner .

```
/
function functionCallWithValue(
    address target,
    bytes memory data,
    uint256 value,
    string memory errorMessage
) internal returns (bytes memory) {
    require(address(this).balance >= value, 'Address: insufficient balance for call');
    return _functionCallWithValue(target, data, value, errorMessage);
}
```

RECOMMENDATION

Project stakeholders should be consulted during the initial asset distribution process.



RECOMMENDATION

Deployer and/or contract owner private keys are secured carefully.

Please refer to PAGE-09 CENTRALIZED PRIVILEGES for a detailed understanding.

ALLEVIATION

The ARBITRUM EXCHANGE project team understands the centralization risk. Some functions are provided privileged access to ensure a good runtime behavior in the project



Identifier	Definition	Severity
COD-10	Third Party Dependencies	Minor 

Smart contract is interacting with third party protocols e.g., Pancakeswap router, cashier contract, protections contract. The scope of the audit treats third party entities as black boxes and assumes their functional correctness. However, in the real world, third parties can be compromised, and exploited. Moreover, upgrades in third parties can create severe impacts, e.g., increased transactional fees, deprecation of previous routers, etc.

RECOMMENDATION

Inspect and validate third party dependencies regularly, and mitigate severe impacts whenever necessary.



CERTIFICATE BY VITAL BLOCK SECURITY



DISCLAIMERS

Vital Block provides the easy-to-understand audit of Solidity, Move and Raw source codes (commonly known as smart contracts).

The smart contract for this particular audit was analyzed for common contract vulnerabilities, and centralization exploits. This audit report makes no statements or warranties on the security of the code. This audit report does not provide any warranty or guarantee regarding the absolute bug-free nature of the smart contract analyzed, nor do they provide any indication of the client's business, business model or legal compliance. This audit report does not extend to the compiler layer, any other areas beyond the programming language, or other programming aspects that could present security risks. Cryptographic tokens are emergent technologies, they carry high levels of technical risks and uncertainty. You agree that your access and/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. This audit report could include false positives, false negatives, and other unpredictable results.

CONFIDENTIALITY

This report is subject to the terms and conditions (including without limitations, description of services, confidentiality, disclaimer and limitation of liability) outlined in the scope of the audit provided to the client. This report should not be transmitted, disclosed, referred to, or relied upon by any individual for any purpose without InterFi Network's prior written consent.

NO FINANCIAL ADVICE

This audit report does not indicate the endorsement of any particular project or team, nor guarantees its security. No third party should rely on the reports in any way, including to make any decisions to buy or sell a product, service or any other asset. The information provided in this report does not constitute investment advice, financial advice, trading advice, or any other sort of advice and you should not treat any of the report's content as such. This audit report should not be used in any way



to make decisions around investment or involvement. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort.

FOR AVOIDANCE OF DOUBT, SERVICES, INCLUDING ANY ASSOCIATED AUDIT REPORTS OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, TAX, LEGAL, REGULATORY, OR OTHER ADVICE.

TECHNICAL DISCLAIMER

ALL SERVICES, AUDIT REPORTS, SMART CONTRACT AUDITS, OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF ARE PROVIDED “AS IS” AND “AS AVAILABLE” AND WITH ALL FAULTS AND DEFECTS WITHOUT WARRANTY OF ANY KIND. TO THE MAXIMUM EXTENT PERMITTED UNDER APPLICABLE LAW, VITAL BLOCK HEREBY DISCLAIMS ALL WARRANTIES, WHETHER EXPRESSED, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO SERVICES, AUDIT REPORT, OR OTHER MATERIALS. WITHOUT LIMITING THE FOREGOING, VITAL BLOCK SPECIFICALLY DISCLAIMS ALL IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT, AND ALL WARRANTIES ARISING FROM THE COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

WITHOUT LIMITING THE FOREGOING, VITAL BLOCK MAKES NO WARRANTY OF ANY KIND THAT ALL SERVICES, AUDIT REPORTS, SMART CONTRACT AUDITS, OR OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF, WILL MEET THE CLIENT’S OR ANY OTHER INDIVIDUAL’S REQUIREMENTS, ACHIEVE ANY INTENDED RESULT, BE COMPATIBLE OR WORK WITH ANY SOFTWARE, SYSTEM, OR OTHER SERVICES, OR BE SECURE, ACCURATE, COMPLETE, FREE OF HARMFUL CODE, OR ERROR-FREE.

TIMELINESS OF CONTENT

The content contained in this audit report is subject to change without any prior notice. Vital Block does not guarantee or warrant the accuracy, timeliness, or completeness of any report you access using the internet or other means, and assumes no obligation to update any information following the publication.



LINKS TO OTHER WEBSITES

This audit report provides, through hypertext or other computer links, access to websites and social accounts operated by individuals other than Vital Block. Such hyperlinks are provided for your reference and convenience only and are the exclusive responsibility of such websites and social accounts owners. You agree that Vital block Security is not responsible for the content or operation of such websites and social accounts and that Vital Block shall have no liability to you or any other person or entity for the use of third-party websites and social accounts. You are solely responsible for determining the extent to which you may use any content at any other websites and social accounts to which you link from the report.



ABOUT VITAL BLOCK

Vital Block provides intelligent blockchain Security Solutions. We provide solidity and Raw Code Review, testing, and auditing services. We have Partnered with 15+ Crypto Launchpads, audited 50+ smart contracts, and analyzed 200,000+ code lines. We have worked on major public blockchains e.g., Ethereum, Binance, Cronos, Doge, Polygon, Avalanche, Metis, Fantom, Bitcoin Cash, Aptos, Oasis, etc.

Vital Block is Dedicated to Making Defi & Web3 A Safer Place. We are Powered by Security engineers, developers, UI experts, and blockchain enthusiasts. Our team currently consists of 5 core members, and 4+ casual contributors.

Website: <https://Vitalblock.org>

Email: info@vitalblock.org

GitHub: <https://github.com/vital-block>

Telegram (Engineering): https://t.me/vital_block

Telegram (Onboarding): https://t.me/vitalblock_cmo





vital-block



info@vitalblock.org



www.Vitalblock.org



Vital Block Dedicated to securing Public and Private Blockchain Ecosystem