Arable

Smart Contract Audit Report

AUDIT SUMMARY

Arable is releasing a series of contracts allowing users to mint synth tokens and earn rewards with those tokens.

For this audit, we reviewed the project team's contracts at commit 7382125ac89a2828d6e45a7260aee65633ab82a3 on the team's GitHub repository.

AUDIT FINDINGS

All findings have been resolved, though some centralized aspects are present.
Date: June 17th, 2022.
Updated: June 22nd, 2022 with updates from commit c3953a617351ba238925dfef86fa19767a18b793 to commit 7382125ac89a2828d6e45a7260aee65633ab82a3.

Finding #1 - ArableCollateral - High (Resolved)

Description: *The decimals used in the maxWithdrawableTokenVal() function and the calculateTokenValue() function are inconsistent when a token does not have 18 decimals.*
Risk/Impact: *If any tokens with less than 18 decimals are used as collateral, users will be able to withdraw deposited collateral without having to burn their minted ArUSD.*
Recommendation: *The calculateTokenValue() function should either divide by the tokens decimals rather than 1 ether or be replaced with the _calculateSingleValueCollateral() function.*
Resolution: *The calculateTokenValue() function now divides based on the token's decimals.*

Finding #2 - ArableCollateral - Medium (Acknowledged)

Description: *The collateral asset allowed rate may be set to 1.*
Risk/Impact: *If the collateral asset allowed rate is too low, users will have no incentive to liquidate an under-collateralized position.*
Recommendation: *The team should consider raising the minimum allowed collateral asset rate.*
Resolution: *The team plans to use asset rates starting at 200%.*

Finding #3 - ArableCollateral - Low (Acknowledged)

Description: *The number of decimals returned from the ArableOracle getPrice() function must be the same number of decimals used in the ArUSD contract.*
Risk/Impact: *As the maxIssuableArUSD() function returns a value in the number of decimals of getPrice(), all amounts of tokens minted and burned are in getPrice() units. Users will be minted and burned the incorrect amount of ArUSD if the decimals are inconsistent.*
Recommendation: *The team must ensure the decimals returned from getPrice() are the same number of decimals as ArUSD. If they are not the same number of decimals, they must multiply or divide the mint and burn amounts by the difference in decimals between the two.*
Resolution: *The team has acknowledged that the getPrice() function returns 18 decimals.*

Finding #4 - ArableCollateral - Informational (Resolved)

Description: *The maxWithdrawableTokenAmount() function is inaccurate in tokens with less than 18 decimals.*

Recommendation: *The maxWithdrawable value should be:*

```
(maxWithdrawableTokenVal(user, token) * 10**tokenDecimals) / tokenPrice
```

Resolution: *The maxWithdrawableTokenAmount() function now divides based on the token's decimals.*

CONTRACTS OVERVIEW

- *The contracts utilize ReentrancyGuard to protect against reentrancy attacks in applicable functions.*
- *As the contracts are implemented with Solidity v0.8.0, they are safe from any possible overflows/underflows.*
- *The team must exercise caution to avoid using fee-on-transfer tokens; if the team does use fee-on-transfer tokens, the contracts must be given proper exclusion from fees.*

ArableAddressRegistry Contract:

- *This contract is used to record the various other contract addresses in the platform.*
- *The owner may update the ArableFarming, ArableOracle, ArableExchange, ArableManager, ArableCollateral, ArableLiquidation, and ArableFeeCollector addresses at any time.*

ArableSynth Contract:

- *The total supply of the token is set to an Admin-defined "total supply limit".*
- *Addresses with the Minter role may mint any amount of tokens up to the total supply limit.*
- *Any user may burn their tokens to reduce the total supply.*
- *Any user can burn tokens on another user's behalf if an allowance has been granted.*
- *The contract complies with the ERC-20 token standard.*
- *Any address with the Admin role may set the total supply limit to any value at any time.*
- *Any address with the Admin role may pause the contract, disabling mint and burn functionality, at any time.*

ArableOracle Contract:

- *Any "allowed provider" address may update the price of any token at any time; updating the price will trigger the "on price change" functionality in the Arable Manager contract.*
- *Allowed providers serve as Oracle Validators within the community and update token prices; the pricing mechanism was outside the scope of this audit so we are unable to give an assessment with regards to security.*
- *Any allowed provider address may update the reward rate for a specified ArableFarming farm and token to any value at any time.*
- *The owner may add and remove any address as an allowed provider at any time.*
- *The owner may pause the contract, disabling the ability to update token price and farm reward rates, at any time.*

ArableManager Contract:

- *When a token price is changed in the ArableOracle contract the on asset price change is triggered; the price difference for the total supply of the token will be added to or removed from the total debt in the Arable Collateral contract.*
- *The owner may deploy and register a new Synth token contract at any time.*
- *The owner may enable and disable any address as a Synth token at any time.*
- *The owner may update the fee model for any asset in the Arable FeeCollector contract at any time.*
- *The owner may pause the contract, preventing the on asset price change functionality, at any time.*

ArableLiquidation Contract:

- *This contract is used to determine if a user's position in the ArableCollateral contract is liquidable.*
- *A user can be flagged for liquidation if their current debt is greater than their max debt or their risk rate is greater than or equal to the liquidation rate.*

- *A user flagged for liquidation is given a liquidation deadline.*
- *Any address may liquidate a flagged user's position.*
- *A user's position may be immediately liquidated if they are greater than or equal to the immediate liquidation rate.*
- *A position that is not past the immediate liquidation rate may not be liquidated until its liquidation deadline has passed.*
- *Liquidating a position will burn ArUSD from the user equivalent to the debt of the position.*
- *The ArableCollateral contract will then liquidate the position. All of the position's debt will be removed and the liquidating user will receive the equivalent amount of collateral with a "liquidation penalty" additional amount, up to the total collateral value for each collateral token in the position.*
- *A user's liquidation flag may be removed if they no longer meet the criteria for liquidation.*
- *The owner may update the AddressRegistry address at any time.*
- *The owner may set the liquidation rate and the immediate liquidation rate to any value at any time.*
- *The owner may set the liquidation delay and liquidation penalty to any value at any time.*
- *The owner may pause the contract at any time, preventing the flagging and unflagging addresses for liquidation, at any time.*

ArableFeeCollector Contract:

- *This contract is used to calculate fees for various tokens and track minter rewards.*
- *Fees are calculated depending on the asset, the current model, and the user who is paying fees.*
- *The asset fees will be the default fees if there is not a fee set for the token in the current model.*
- *The "account bonus" for the user will be the default account bonus if there is not one set for the address and token.*
- *The fees paid will be the difference between the account bonus and asset fees as a percentage.*
- *Collected fees will be transferred to the ArableExchange where they are converted to ArUSD.*
- *Any "allowed provider" may increase minter rewards for any address by any amount once per epoch and reward token.*
- *Addresses that have been allocated rewards may claim their rewards tokens at any time given their risk rate in the ArableCollateral contract is not greater than 100%; the contract must have sufficient tokens or the rewards claim will fail.*
- *Any allowed provider address may start a new epoch once the current epoch has passed.*
- *The ArableAddressRegistry may set the fees for any asset and model at any time.*
- *The owner may add and remove an address as an allowed provider at any time.*
- *The owner may set the reward tokens at any time; doing so will remove all existing tokens.*
- *The owner may remove all reward tokens at any time.*
- *The owner may pause the contract, preventing a new epoch, paying fees, and rewards allocation and claiming, at any time.*

ArableFarming Contract:

- *Any user may deposit specified tokens into non-disabled farms to earn rewards in potentially multiple tokens.*
- *Users will earn rewards per epoch they are staking tokens.*
- *Any address may update the rewards rate sum used to calculate users' rewards from a farm.*
- *Rewards may be updated once per epoch and up to 10 epochs at a time.*
- *Users may unstake their tokens from any non-disabled farms at any time.*
- *Users will pay fees when depositing and withdrawing determined in the ArableFeeCollector contract.*
- *Users may claim their rewards at any time.*
- *The owner may add a new farm and its specified staking token at any time.*
- *The owner may set the rewards tokens for any farm at any time.*
- *The owner may remove all rewards tokens from a farm at any time.*
- *The owner may enable and disable a farm at any time.*
- *The owner may pause the contract, preventing staking, unstaking, and rewards claims, at any time.*

ArableExchange Contract:

- *This contract is used to convert between synth tokens.*
- *The FeeCollector contract may use this contract to convert a non-disabled synth token to ArUSD.*
- *The price for the token being converted is fetched from the ArableOracle contract.*
- *The specified amount of the converted token will be burned and the corresponding value in ArUSD will be minted to the FeeCollector contract.*
- *Any address may use this contract to convert between any two non-disabled synth tokens.*
- *The price for each token is fetched from the ArableOracle contract.*

- *Users will pay a fee determined in the ArableFeeCollector contract.*
- *The specified amount of the token being converted will be burned from the user and they will be minted the corresponding value of the desired token.*
- *The owner may pause the contract, preventing any conversions, at any time.*

StabilityFund Contract:

- *Any user may deposit a non-disabled stable token into the contract; in return, they will receive tokens representing the value they have contributed to the contract.*
- *Users may withdraw their tokens at any time; this will burn their StabilityFund tokens and they will receive the corresponding amount of all stable tokens in the contract.*
- *The team must ensure that the price difference between any two tokens does not become too high; if the price difference becomes too high, an arbitrage opportunity will be present.*
- *When enabled, users may perform a 1-to-1 swap between any two non-disabled stable tokens.*
- *Users will pay a swap fee which will remain in the contract and be distributed when users withdraw.*
- *The contract must have a sufficient amount of the desired token or the swap will fail.*
- *Users who have been added to the flash loans whitelist may perform a flash loan with a non-disabled token at any time.*
- *Users will receive the specified amount of tokens from the contract, which will call the executeOperation() function on the msg sender.*
- *The msg sender must then use the flash repay functionality to return an equal or greater amount of tokens in the same transaction.*
- *The owner may add any address as a stable token at any time.*
- *The owner may remove any added stable token address given that the contract has none of the token.*
- *The owner may toggle whether a token is disabled at any time.*
- *The owner may toggle whether swaps are enabled at any time.*
- *The owner may toggle whether an address may perform a flash loan at any time.*
- *The owner may withdraw any token that was not set as a stable token at any time.*
- *The owner may pause the contract, disabling deposits, withdrawals, swaps, and flash loans, at any time.*

ArableCollateral Contract:

- *Any user may deposit collateral in the contract. Deposited collateral will be added to the user's individual collateral and will remove their liquidation flag in the ArableLiquidation contract if appropriate.*
- *Any user may withdraw their deposited collateral up to the amount that the user would remain collateralized.*
- *Users may mint ArUSD up to the maximum issuable ArUSD amount.*
- *Minting will increase a user's current debt factor and the total debt factor. A user's real debt value is determined from their debt factor times the ratio of the total debt factor and the total debt.*
- *A user's maximum issuable ArUSD is determined from the total value of all of their deposited collateral tokens and the allowed collateral rate of each deposited collateral token.*
- *Users may burn ArUSD up to their current total debt. The user's liquidation flag in the ArableLiquidation contract may be removed if applicable.*
- *The ArableLiquidation contract may liquidate a user's position that has been flagged at any time.*
- *The ArableManager and ArableFarming contracts may add and remove from the total debt whenever a token's price is updated. This will increase or decrease all users' debt as a result.*
- *The owner may pause the contract, disabling withdrawing collateral and minting and burning ArUSD, at any time.*
- *The owner may update the AddressRegistry address at any time.*
- *The owner may add and remove an address as a collateral token at any time.*
- *The owner may change the allowed rate for any collateral token at any time.*

AUDIT RESULTS

Vulnerability Category	Notes	Result
Arbitrary Jump/Storage Write	N/A	PASS
Centralization of Control	• The team may update any address in the ArableAddressRegistry. • Allowed Providers may set the price of any token to any value in the ArableOracle. • All contracts may be paused preventing certain functionality within each contract.	WARNING
Compiler Issues	N/A	PASS
Delegate Call to Untrusted Contract	N/A	PASS
Dependence on Predictable Variables	N/A	PASS
Ether/Token Theft	N/A	PASS
Flash Loans	N/A	PASS
Front Running	N/A	PASS
Improper Events	N/A	PASS
Improper Authorization Scheme	N/A	PASS
Integer Over/Underflow	N/A	PASS
Logical Issues	N/A	PASS
Oracle Issues	N/A	PASS
Outdated Compiler Version	N/A	PASS
Race Conditions	N/A	PASS
Reentrancy	N/A	PASS
Signature Issues	N/A	PASS
Unbounded Loops	N/A	PASS
Unused Code	N/A	PASS
Overall Contract Safety		PASS

CONTRACT SOURCE SUMMARY AND VISUALIZATIONS

Name	Address/Source Code	Visualized (Hover-Zoom Recommended)
ArableAddressRegistry	GitHub (Not yet deployed on mainnet)	Inheritance Chart. Function Graph.
ArableCollateral	GitHub (Not yet deployed on mainnet)	Inheritance Chart. Function Graph.
ArableExchange	GitHub (Not yet deployed on mainnet)	Inheritance Chart. Function Graph.
ArableFarming	GitHub (Not yet deployed on mainnet)	Inheritance Chart. Function Graph.
ArableFeeCollector	GitHub (Not yet deployed on mainnet)	Inheritance Chart. Function Graph.
ArableLiquidation	GitHub (Not yet deployed on mainnet)	Inheritance Chart. Function Graph.
ArableManager	GitHub (Not yet deployed on mainnet)	Inheritance Chart. Function Graph.
ArableOracle	GitHub (Not yet deployed on mainnet)	Inheritance Chart. Function Graph.
ArableSynth	GitHub (Not yet deployed on mainnet)	Inheritance Chart. Function Graph.
StabilityFund	GitHub (Not yet deployed on mainnet)	Inheritance Chart. Function Graph.

ABOUT SOLIDITY FINANCE

Solidity Finance was founded in 2020 and quickly grew to have one of the most experienced and well-equipped smart contract auditing teams in the industry. Our team has conducted 1300+ solidity smart contract audits covering all major project types and protocols, securing a total of over $10 billion U.S. dollars in on-chain value.
Our firm is well-reputed in the community and is trusted as a top smart contract auditing company for the review of solidity code, no matter how complex. Our team of experienced solidity smart contract auditors performs audits for tokens, NFTs, crowdsales, marketplaces, gambling games, financial protocols, and more!

Contact us today to get a free quote for a smart contract audit of your project!

WHAT IS A SOLIDITY AUDIT?

Typically, a smart contract audit is a comprehensive review process designed to discover logical errors, security vulnerabilities, and optimization opportunities within code. A *Solidity Audit* takes this a step further by verifying economic logic to ensure the stability of smart contracts and highlighting privileged functionality to create a report that is easy to understand for developers and community members alike.

HOW DO I INTERPRET THE FINDINGS?

Each of our Findings will be labeled with a Severity level. We always recommend the team resolve High, Medium, and Low severity findings prior to deploying the code to the mainnet. Here is a breakdown on what each Severity level means for the project:

- **High** severity indicates that the issue puts a large number of users' funds at risk and has a high probability of exploitation, or the smart contract contains serious logical issues which can prevent the code from operating as intended.
- **Medium** severity issues are those which place at least some users' funds at risk and has a medium to high probability of exploitation.
- **Low** severity issues have a relatively minor risk association; these issues have a low probability of occurring or may have a minimal impact.
- **Informational** issues pose no immediate risk, but inform the project team of opportunities for gas optimizations and following smart contract security best practices.