Collar Quest Auction & ICO

Smart Contract Audit Report

AUDIT SUMMARY

Collar Quest is releasing an auction platform and ICO contract for NFTs.

For this audit, we reviewed Collar Quest's SPARCEClockAuction contract at 0xe499cee42868a0b4f8ac2152a77b012530b361e4 and InitialCoinOffering contract at 0xf02c7673735f7c5921b4530ece6790b2449d6cba on the Ethereum Mainnet.

We previously reviewed the project team's SPARCE contract here and Battle and Breeding contracts here.

AUDIT FINDINGS

High findings were identified and the team must resolve these issues. In addition, centralized aspects are present.
Date: September 29th, 2022.

Finding #1 - SPARCEClockAuction - High

Description: *The createAuction() function relies on the SPARCECore NFT contract's getSparce() function for NFT data.*
Risk/Impact: *The getSparce() function will revert if an NFT ID is provided that does not exist in the SPARCECore NFT contract. As a result, NFTs from other NFT collections will be rejected if their ID does not also exist in the SPARCECore NFT contract.*
Recommendation: *The team should restructure the call to the getSparce() function as follows:*

```
address _operator;
bool _newlyBorn;
if(_nftAddress == address(collarQuest) {
    (, , _operator, _newlyBorn ) =  collarQuest.getSparce(
}
```

Resolution: *The team has not yet addressed this issue.*

Finding #2 - SPARCEClockAuction - High

Description: *The calculateSaleRoyalty() function will return the default variable values if the NFT is not a SPARC-E NFT.*

Risk/Impact: *Any non-SPARC-E NFT will have a price of 0 regardless of the price set by the user.*

Recommendation: *The team should set the _price value before returning:*

```
if(_nftAddress != address(collarQuest)) {
    _price = price;
    return (_royalityFee,_daoFee,_price);
}
```

Resolution: *The team has not yet addressed this issue.*

Finding #3 - SPARCEClockAuction - High

Description: *The contract contains no external functions to toggle the paused and unpaused state.*

Risk/Impact: *The whenPaused() modifier will have no effect.*

Recommendation: *The team should add external owner restricted pause() and unpause() functions.*

Resolution: *The team has not yet addressed this issue.*

Finding #4 - InitialCoinOffering - Medium

Description: *The contract relies on the code.length value to determine if an address is a contract in the _isContract() modifier:*

```
function _isContract(address account) private view returns
    return account.code.length > 0;
}
```

Risk/Impact: *A contract could call these functions in its constructor to bypass this restriction.*

Recommendation: *The team should rather enforce that the msg.sender is the same as the tx.origin in the _isContract() modifier:*

```
function _isContract(address account) private view returns
    return msg.sender == tx.origin;
}
```

Resolution: *The team has not yet addressed this issue.*

Finding #5 - InitialCoinOffering - Low

Description: *The currentSaleId value is incremented after the public sale start and end times are set.*

Risk/Impact: *Whitelisted users could reach the max buy value twice if the presale is open before the public sale start and end times are set.*

Recommendation: *The team should should consider using a*

single integer buy count rather than a mapping.

Resolution: *The team has not yet addressed this issue.*

Finding #6 - SPARCEClockAuction - Low

Description: *The contract uses the native .transfer() function to send ETH.*

Risk/Impact: *Contracts that are the recipient of ETH transfers using the .transfer() function will be unable to perform any non-trivial functionality in their receive() or fallback() functions.*

Recommendation: *The team should use the .call() function instead of the .transfer() function.*

Resolution: *The team has not yet addressed this issue.*

Finding #7 - InitialCoinOffering - Informational

Description: *The public sale start time can be set to a timestamp in the past.*

Recommendation: *The team should enforce that the public start time is greater than the block.timestamp.*

Resolution: *The team has not yet addressed this issue.*

Finding #8 - SPARCEClockAuction - Informational

Description: *The royalty and DAO fee setters only check the individual tax amounts are less than the DIVISOR value.*
Recommendation: *The team should should consider enforcing that the sum of the royalty and DAO fees are less than the DIVISOR.*
Resolution: *The team has not yet addressed this issue.*

CONTRACTS OVERVIEW

- *As the contracts are implemented with Solidity v0.8.4, they are safe from any possible overflows/underflows.*

InitialCoinOffering Contract:

- *This contract allows the owner to list SPARC-E NFTs for sale in various initial offering stages.*
- *The sale is split into presale, public sale, and crowd sale and will occur in that order chronologically.*
- *The owner may list any number of NFTs for sale at any price at any time.*
- *The owner may remove any NFT from sale at any time.*
- *The owner may update the price of any NFT for sale at any time.*
- *Whitelisted EOAs may purchase may purchase NFTs during the*

presale period.

- *NFTs purchased in this period will receive the presale level of discount.*
- *Users may only buy up to the "max buy" amount of NFTs in this period.*
- *Any EOA may purchase any number of NFTs during the public sale period.*
- *NFTs purchased in this period will receive the public sale level of discount.*
- *Any EOA may purchase any number of NFTs during the crowd sale period after the public sale has ended.*
- *NFTs purchased in this period will receive the crowd sale level of discount.*
- *The owner may set the presale start and end time to any values in the future given that the public sale times have not been set.*
- *The owner may set the public sale start and end time to any values after the presale end time once the presale times have been set.*
- *The owner may reset the presale and public sale timestamps if the public sale end time has not passed.*
- *The owner may update the discount for each stage at any time.*
- *The owner may update the max buy amount to any value at any time.*
- *The owner may update the SPARC-E NFT address at any time.*
- *The owner may add and remove any address from the whitelist*

at any time.

- *The owner may withdraw any excess ETH in the contract at any time.*
- *The owner may pause the contract, preventing the purchase of NFTs, at any time.*
- *The contract utilizes ReentrancyGuard to protect against reentrancy attacks in applicable functions.*

SPARCEClockAuction Contract:

- *This contract allows users to list NFTs for sale.*
- *Any user may list an NFT for sale.*
- *The contract is intended to allow NFTs from any NFT collection to be listed for sale, however, due to logical issues only NFT IDs that exist in the SPARCECore contract can be listed.*
- *Users must specify a starting price, ending price, and auction duration.*
- *An NFT's price will change over the sale duration increasing or decreasing until it reaches the end price.*
- *The auction duration must be at least 1 minute long.*
- *Users may bid on a listed NFT by providing a message signed by the Signer address.*
- *If the bid amount is at least the NFT's current price, the user will successfully buy the token.*
- *Due to logical issues the price for non-SPARC-E NFTs will be zero.*

- *An additional royalty and DAO fee may be taken if the contract is a SPARC-E NFT.*
- *The fees are dependent upon whether the SPARC-E NFT is "newly born" and if the Operator associated with the NFT is the Breeding contract.*
- *The Treasury address will be sent the DAO fee, the Royalty address will be sent the royalty fee, if applicable, and the remaining ETH sent to the NFT seller.*
- *Any excess ETH will be returned to the buyer.*
- *If a SPARC-E NFT was newly born when purchased, it will no longer be newly born.*
- *The owner of a listing may cancel the listing at any time.*
- *The owner may cancel any listing when the contract is paused.*
- *The owner may update the Breeding contract address, Signer address, Treasury address, and SPARCECore NFT address at any time.*
- *The owner may update the fees for Breeder and non-Breeder sales and newly born and not newly born sales to any value up to 100% at any time.*

AUDIT RESULTS

Vulnerability Category	Notes	Result

Arbitrary Jump/Storage Write	N/A	PASS
Centralization of Control	• The SPARCEClockAuction owner may cancel all listings if the contract is paused. • The SPARCEClockAuction owner may set auction fees up to 100%. • The InitialCoinOffering owner may reset the presale and public sale start and end times.	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	• Only NFT IDs that exist in the SPARCECore contract may be listed. • Non-SPARC-E NFTs will always have a price of 0. • The contract contains no external functions to toggle the paused state. • The InitialCoinOffering	FAIL

	contract relies on code.length which can be bypassed in a constructor.	
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		FAIL

InitialCoinOffering Contract

INHERITANCE CHART

FUNCTION GRAPH

FUNCTIONS OVERVIEW

SPARCEClockAuction Contract

INHERITANCE CHART

FUNCTION GRAPH

FUNCTIONS OVERVIEW

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.

GO HOME

© Solidity Finance LLC. | All rights reserved.
Please note we are not associated with the Solidity programming language or the core team which develops the language.

Please review our Terms & Conditions and Privacy Policy. By using this site, you agree to these terms.