QuillAudits

Audit Report
September, 2021

Contents

Scope of Audit

The scope of this audit was to analyze and document the Avatar Art Market smart contract codebase for quality, security, and correctness.

Checked Vulnerabilities

We have scanned the smart contract for commonly known and more specific vulnerabilities. Here are some of the commonly known vulnerabilities that we considered:

- Re-entrancy
- Timestamp Dependence
- Gas Limit and Loops
- DoS with Block Gas Limit
- Transaction-Ordering Dependence
- Use of tx.origin
- Exception disorder
- Gasless send
- Balance equality
- Byte array
- Transfer forwards all gas
- ERC20 API violation
- Malicious libraries
- Compiler version not fixed
- Redundant fallback function
- Send instead of transfer
- Style guide violation
- Unchecked external call
- Unchecked math
- Unsafe type inference
- Implicit visibility level

Techniques and Methods

Throughout the audit of smart contract, care was taken to ensure:

- The overall quality of code.
- Use of best practices.
- Code documentation and comments match logic and expected behaviour.
- Token distribution and calculations are as per the intended behaviour mentioned in the whitepaper.
- Implementation of ERC-20 token standards.
- Efficient use of gas.
- Code is safe from re-entrancy and other vulnerabilities.

The following techniques, methods and tools were used to review all the smart contracts.

Structural Analysis
In this step we have analyzed the design patterns and structure of smart contracts. A thorough check was done to ensure the smart contract is structured in a way that will not result in future problems.
SmartCheck.

Static Analysis
Static Analysis of Smart Contracts was done to identify contract vulnerabilities. In this step a series of automated tools are used to test security of smart contracts.

Code Review / Manual Analysis
Manual Analysis or review of code was done to identify new vulnerability or verify the vulnerabilities found during the static analysis. Contracts were completely manually analyzed, their logic was checked and compared with the one described in the whitepaper. Besides, the results of automated analysis were manually verified.

Gas Consumption
In this step we have checked the behaviour of smart contracts in production. Checks were done to know how much gas gets consumed and possibilities of optimization of code to reduce gas consumption.

Tools and Platforms used for Audit
Remix IDE, Truffle, Truffle Team, Ganache, Solhint, Mythril, Slither, SmartCheck.

Issue Categories

Every issue in this report has been assigned with a severity level. There are four levels of severity and each of them has been explained below.

High severity issues

A high severity issue or vulnerability means that your smart contract can be exploited. Issues on this level are critical to the smart contract's performance or functionality and we recommend these issues to be fixed before moving to a live environment.

Medium level severity issues

The issues marked as medium severity usually arise because of errors and deficiencies in the smart contract code. Issues on this level could potentially bring problems and they should still be fixed.

Low level severity issues

Low level severity issues can cause minor impact and or are just warnings that can remain unfixed for now. It would be better to fix these issues at some point in the future.

Informational

These are severity four issues which indicate an improvement request, a general question, a cosmetic or documentation error, or a request for information. There is low-to-no impact.

Number of issues per severity

Type	High	Medium	Low	Informational
Open	0	0	0	0
Acknowledged	1	1	7	11
Closed	2	1	5	4

Introduction

During the period of **September 9th, 2021 to September 13th, 2021** - QuillAudits Team performed a security audit for Avatar Art Market smart contracts.

The code for the audit was taken from following the official links: https://github.com/bytenext/avatar-art-market/tree/main/contracts

Note	Date	Commit hash
Version 1	09/09/2021	83897b4075720fe76524d3face58cfebd6bfe667
Version 2	13/09/2021	7b2b844c0c02451d77ee3c8cb5de7d7bfc358a6a

Issues Found – Code Review / Manual Testing

A. Contract - AvatarArtArtistKYC.sol

High severity issues

No issues were found.

Medium severity issues

No issues were found.

Low level severity issues

No issues were found.

Informational

1. **Unnecessary code**

   Description
   Unnecessary interface IAvatarArtArtistKYC at line number #11.

   Remediation
   Remove unnecessary code to make code clean.

   Status: Fixed
   This issue was reported in Version 1 and found fixed in Version 2.

2. **Use the latest solidity version**

   Description
   Using the latest solidity will prevent any compiler-level bugs.

   Remediation
   Please use 0.8.7 which is the latest version.

   Status: Acknowledged by the Auditee

B. Contract - AvatarArtNFT.sol

High severity issues

No issues were found.

Medium severity issues

1. **Owner can burn anyone's token**

Description
burn() function at line #26 allows the owner to burn anyone's token. Ideally the token holder must be able to burn his own tokens and not the contract owner. Otherwise, users may feel fearful that the owner can burn his tokens anytime. Or, if the private key of the owner's wallet is compromised, then their own tokens also will be at risk of being burnt.

Remediation
Best approach is that the token holder should be able to burn his own tokens and not the contract owner.

Status: Fixed
This issue was reported in Version 1 and found fixed in Version 2.

Low level severity issues

2. **Critical operation lacks event log**

Description
Missing event log for : setAvatarArtArtistKYC.

Remediation
Please write an event log for listed events.

Status: Acknowledged by the Auditee

Informational

3. Use the latest solidity version

Description
Using the latest solidity will prevent any compiler-level bugs.

Remediation
Please use 0.8.7 which is the latest version.

Status: Acknowledged by the Auditee

C. Contract - AvatarArtAuction.sol

High severity issues

1. Infinite loops

Description
As array elements will increase, then it will cost more and more gas. And eventually, it will stop all the functionality. After several hundreds of trades orders, all those functions depending on it will stop. We suggest avoiding loops. For example, use mapping to store the array index. And query that data directly, instead of looping through all the elements to find an element. If you need more clarification on this, please let us know and we will discuss it in detail.

Remediation
Adjust logic to replace loops with mapping or other code structure.

Status: Fixed
This issue was reported in Version 1 and found fixed in Version 2.

Medium severity issues

No issues were found.

Low level severity issues

2. Owner has access to all BNU tokens

Description
The contract owner can drain the BNU tokens from the contract. People might question the owner for his ability to rugpull. If this is a desired feature, then please ignore this point.

Status: Acknowledged by the Auditee

3. Critical operation lacks event log

Description
Missing event log for : withdrawToken, deactivateAuction.

Remediation
Please write an event log for listed events.

Status: Fixed
This issue was reported in Version 1 and found fixed in Version 2.

Informational

4. Use the latest solidity version

Description
Using the latest solidity will prevent any compiler-level bugs.

Remediation
Please use 0.8.7 which is the latest version.

Status: Acknowledged by the Auditee

5. Missing error message in require condition

Description
It is best practice to add custom error messages in every required condition, which would be helpful in debugging as well as it will give clear indication to any transaction failure.

Remediation
Add custom error messages in every required condition.

Status: Fixed
This issue was reported in Version 1 and found fixed in Version 2.

6. Deactivated auction can not be reactivated again

Description
Once any auction is deactivated by the owner, it can not be reactivated again. He has to put in new ones. If this is desired behavior, then this point can be safely ignored.

Status: Acknowledged by the Auditee

7. Spelling mistake

Description
Line number #225 has "updateActionPrice" instead of "updateAuctionPrice". Same way at line number #239. Although this does not raise any security or logical issues, correct spelling will increase readability.

Remediation
Correct spelling mistakes.

Status: Fixed
This issue was reported in Version 1 and found fixed in Version 2.

8. Unnecessary code

Description
Unnecessary inheritance IAvatarArtAuction at line number #20. it does not serve any purpose.

Remediation
Remove unnecessary code to make code clean.

Status: Fixed
This issue was reported in Version 1 and found fixed in Version 2.

D. Contract - AvatarArtBase.sol

High severity issues

No issues were found.

Medium severity issues

No issues were found.

Low level severity issues

No issues were found.

Informational

1. **Use the latest solidity version**

   Description
   Using the latest solidity will prevent any compiler-level bugs.

   Remediation
   Please use 0.8.7 which is the latest version.

   Status: **Acknowledged by the Auditee**

E. Contract - AvatarArtMarketplace.sol

High severity issues

1. **Infinite loops**

   Description
   As array elements will increase, then it will cost more and more gas. And eventually, it will stop all the functionality. After several hundreds of trades orders, all those functions depending on it will stop. We suggest avoiding loops. For example, use mapping to store the array index. And query that data directly, instead of looping through all the elements to find an element. If you need more clarification on this, please let us know and we will discuss it in detail.

   Remediation
   Adjust logic to replace loops with mapping or other code structure.

Status: Fixed
This issue was reported in Version 1 and found fixed in Version 2.

Medium severity issues

No issues were found.

Low level severity issues

2. Critical operation lacks event log

Description
Missing event log for: cancelSellOrder, withdrawToken, _removeFromTokens.

Remediation
Please write an event log for listed events.

Status: Fixed
This issue was reported in Version 1 and found fixed in Version 2.

3. Unnecessary extra token transfer

Description
purchase() function at line number #115 has 3 token transfers. Here, if we transfer directly from msg.sender to owner and contract directly, then it will save one token transfer, saving gas cost.

Status: Fixed
This issue was reported in Version 1 and found fixed in Version 2.

4. Owner role can be delegated to sub-admin

Description
createSellOrder() function at line #43 usually can be called from the server side scripts to automate the process. In that case, the owner's main private key needs to be put in the server script. Which is a security threat. Usually, there should be a sub-admin wallet which performs this action. and the owner can change this anytime if that wallet is ever compromised.

Remediation
have a sub-admin wallet for createSellOrder function only, which can be changed by the owner.

Status: Fixed
This issue was reported in Version 1 and found fixed in Version 2.

Informational

5. Use the latest solidity version

Description
Using the latest solidity will prevent any compiler-level bugs.

Remediation
Please use 0.8.7 which is the latest version.

Status: Acknowledged by the Auditee

6. Function input parameters lack of check

Description
Variable validation is not performed in below functions :
withdrawToken = tokenAddress | purchase = tokenId | cancelSellOrder = tokenId | createSellOrder = tokenId , price.

Remediation
Variable should not be empty and > 0 for address type check variable is not address(0).

Status: Acknowledged by the Auditee

F. Contract - BNUToken.sol

High severity issues

No issues were found.

Medium severity issues

No issues were found.

Low level severity issues

No issues were found.

Informational

1. **Use the latest solidity version**

   Description
   Using the latest solidity will prevent any compiler-level bugs.

   Remediation
   Please use 0.8.7 which is the latest version.

   Status: Acknowledged by the Auditee

G. Contract - AvatarArtERC20.sol

High severity issues

No issues were found.

Medium severity issues

No issues were found.

Low level severity issues

1. **Unlimited minting**

   Description
   Owner can do unlimited minting. This is not a good practice for healthy tokenomics. On another hand, please double confirm the logic. If this is a requirement of the business plan, then ok. Otherwise It needs to be removed or adjusted.

   Status: Acknowledged by the Auditee

2. **Owner can burn anyone's token**

   Description
   Burn function at line number #18 allows the owner to burn anyone's tokens. It is good practice that only token holders should be able to burn their own tokens.

   Status: Acknowledged by the Auditee

Informational

3. Use the latest solidity version

Description
Using the latest solidity will prevent any compiler-level bugs.

Remediation
Please use 0.8.7 which is the latest version.

Status: Acknowledged by the Auditee

H. Contract - AvatarArtExchange.sol

High severity issues

1. Infinite loops

Description
As array elements will increase, then it will cost more and more gas. And eventually, it will stop all the functionality. After several hundreds of trades orders, all those functions depending on it will stop. We suggest avoiding loops. For example, use mapping to store the array index. And query that data directly, instead of looping through all the elements to find an element. If you need more clarification on this, please let us know and we will discuss it in detail.

Remediation
Adjust logic to replace loops with mapping or other code structure.

Status: Acknowledged by the Auditee

Medium severity issues

No issues were found.

Low level severity issues

2. Critical operation lacks event log

Description
Missing event log for ： setFee, setPairInfo.

Remediation
Please write an event log for listed events.

Status: Acknowledged by the Auditee

3. Owner can drain all ERC20 tokens

Description
The function withdrawToken() at line number #458 will allow the owner to withdraw all the ERC20 tokens. This would create trust issues in the users. If these are the desired features, then please ignore this point.

Status: Acknowledged by the Auditee

Informational

4. Use the latest solidity version

Description
Using the latest solidity will prevent any compiler-level bugs.

Remediation
Please use 0.8.7 which is the latest version.

Status: Acknowledged by the Auditee

I. Contract - AvatarArtTokenDeployer.sol

High severity issues

No issues were found.

Medium severity issues

1. User might not have all the tokens

Description
The condition at line number #90 can potentially break the burnToken() function. There are many scenarios, where even a fraction of AvatarArtERC20 token is sent to an unintended wallet or dead wallet, then this function will never execute for that particular token.

Remediation
Double confirm the business plan requirement. And if this is not a necessary requirement, then adjust the logic accordingly.

Status: Acknowledged by the Auditee

Low level severity issues

2. Owner can drain all NFTs

Description
If this is part of the business plan, then please ignore this point. But an over-power role creates trust issues in the community.

Status: Acknowledged by the Auditee

3. Critical operation lacks event log

Description
Missing event log for ： approve, withdrawNft, toggleAllowedPair.

Remediation
Please write an event log for listed events.

Status: Fixed
This issue was reported in Version 1 and found fixed in Version 2.

Informational

4. Use the latest solidity version

Description
Using the latest solidity will prevent any compiler-level bugs.

Remediation
Please use 0.8.7 which is the latest version.

Status: Acknowledged by the Auditee

Functional test

File: AvatarArtArtistKYC.sol

Function Names	Testing results
isVerified	Passed
toggleVerify	Passed
onlyOwner	Passed
renounceOwnership	Passed
transferOwnership	Passed
_setOwner	Passed

File: AvatarArtNFT.sol

Function Names	Testing results
constructor	Passed
create	Passed
burn	Passed
getAvatarArtArtistKYC	Passed
setAvatarArtArtistKYC	Critical operation lacks event log
_baseURI	Passed
onlyOwner	Passed
renounceOwnership	Passed
transferOwnership	Passed
_setOwner	Passed
supportsInterface	Passed

balanceOf	Passed
ownerOf	Passed
name	Passed
symbol	Passed
tokenURI	Passed
_baseURI	Passed
approve	Passed
getApproved	Passed
setApprovalForAll	Passed
isApprovedForAll	Passed
transferFrom	Passed
safeTransferFrom	Passed
_safeTransfer	Passed
_exists	Passed
_isApprovedOrOwner	Passed
_safeMint	Passed
_mint	Passed
_burn	Passed
_transfer	Passed
_approve	Passed
_checkOnERC721Received	Passed
_beforeTokenTransfer	Passed

File : AvatarArtAuction.sol

Function Names	Testing results
constructor	Passed
getAuctionCount	Passed
createAuction	Passed
deactivateAuction	Passed
distribute	Passed
getActiveAuctionByTokenId	Passed
getAuction	Passed
getAuctionWinnersByTokenId	Passed
place	Passed
updateActionPrice	Passed
updateActionTime	Passed
withdrawToken	Passed
getBnuToken	Passed
getAvatarArtNFT	Passed
getFeePercent	Passed
setAvatarArtNFT	Passed
setBnuToken	Passed
setFeePercent	Passed
onERC721Received	Passed

File : AvatarArtBase.sol

Function Names	Testing results
getBnuToken	Passed
getAvatarArtNFT	Passed
getFeePercent	Passed
setAvatarArtNFT	Passed
setBnuToken	Passed
setFeePercent	Passed
onERC721Received	Passed
onlyOwner	Passed
renounceOwnership	Passed
transferOwnership	Passed
_setOwner	Passed

File: AvatarArtMarketplace.sol

Function Names	Testing results
getBnuToken	Passed
getAvatarArtNFT	Passed
getFeePercent	Passed
setAvatarArtNFT	Passed
setBnuToken	Passed
setFeePercent	Passed
onERC721Received	Passed

constructor	Passed
cancelSellOrder	Passed
getTokens	Passed
getTokenInfo	Passed
getMarketHistories	Passed
getTokenPrice	Passed
getTokenOwner	Passed
purchase	Passed
withdrawToken	Passed
_removeFromTokens	Removed

File: BNUToken.sol

Function Names	Testing results
onlyOwner	Passed
renounceOwnership	Passed
transferOwnership	Passed
_setOwner	Passed
constructor	Passed
mint	Passed
burn	Passed

File: AvatarArtERC20.sol

Function Names	Testing results
onlyOwner	Passed
renounceOwnership	Passed
transferOwnership	Passed
_setOwner	Passed
constructor	Passed
name	Passed
symbol	Passed
decimals	Passed
totalSupply	Passed
balanceOf	Passed
transfer	Passed
allowance	Passed
approve	Passed
transferFrom	Passed
increaseAllowance	Passed
decreaseAllowance	Passed
_transfer	Passed
_mint	Passed
_burn	Passed
_approve	Passed

_beforeTokenTransfer	Passed
_afterTokenTransfer	Passed
mint	Passed
burn	Owner can burn anyone's token

File : AvatarArtExchange.sol

Function Names	Testing results
onlyOwner	Passed
renounceOwnership	Passed
transferOwnership	Passed
_setOwner	Passed
isRunning	Passed
constructor	Passed
toggleRunningStatus	Passed
getRunningStatus	Passed
getOpenOrders	Passed
getOpenBuyOrdersForPrice	Passed
getOrders	Passed
getUserOrders	Passed
getOpenSellOrdersForPrice	Passed
isTradable	Passed
setFee	Critical operation lacks event log
setPairInfo	Critical operation lacks event log

buy	Passed
sell	Passed
cancel	Passed
withdrawFee	Passed
withdrawToken	Owner can drain all ERC20 tokens
_increaseFeeReward	Passed
_cancelBuyOrder	Passed
_cancelSellOrder	Passed
_increaseFilledQuantity	Passed
_updateOrderToBeFilled	Passed

Automated Testing

Automated Testing results can be found here.

Closing Summary

Overall, smart contracts are very well written and adhere to guidelines.

No instances of Integer Overflow and Underflow vulnerabilities or Back-Door Entry were found in the contract, but relying on other contracts might cause Reentrancy Vulnerability.

The majority of the concerns addressed above have been acknowledged, implemented and verified.

Disclaimer

Quillhash audit is not a security warranty, investment advice, or an endorsement of the Avatar-Art-Market platform. This audit does not provide a security or correctness guarantee of the audited smart contracts. The statements made in this document should not be interpreted as investment or legal advice, nor should its authors be held accountable for decisions made based on them. Securing smart contracts is a multistep process. One audit cannot be considered enough. We recommend that the Avatar Art Market team put in place a bug bounty program to encourage further analysis of the smart contract by other third parties.

QuillAudits

Canada, India, Singapore and United Kingdom

audits.quillhash.com

audits@quillhash.com