0x Guard

Smart contracts security assessment

Final report

Tariff: Standard

Quartz Vault

March 2022

0xguard.com

hello@0xguard.com

Contents

Introduction

Beefy fork contract.

The code is available in the Github repository.

The code was checked after 7bca501 commit.

Users should check that they interact with the same contracts that were been audited.

Name	Quartz Vault
Audit date	2022-03-04 - 2022-03-05
Language	Solidity
Platform	Binance Smart Chain

Contracts checked

Name	Address
Multiple contract	
StrategyQuartzLP	https://github.com/0xBriz/quartz-vaults/blob/7bca5011298d463b5af992eefc6074a58c1004be/flattened/StrategyQuartzLPFlattened.sol

Procedure

We perform our audit according to the following procedure:

Automated analysis

- Scanning the project's smart contracts with several publicly available automated Solidity analysis tools
- Manual verification (reject or confirm) all the issues found by the tools

Manual audit

- Manually analyse smart contracts for security vulnerabilities
- Smart contracts' logic check

Known vulnerabilities checked

Title	Check result
Unencrypted Private Data On-Chain	passed
Code With No Effects	passed
Message call with hardcoded gas amount	passed
Typographical Error	passed
DoS With Block Gas Limit	passed
Presence of unused variables	passed
Incorrect Inheritance Order	passed
Requirement Violation	passed
Weak Sources of Randomness from Chain Attributes	passed
Shadowing State Variables	passed
Incorrect Constructor Name	passed
Block values as a proxy for time	passed
Authorization through tx.origin	passed
DoS with Failed Call	passed
Delegatecall to Untrusted Callee	passed
Use of Deprecated Solidity Functions	passed
Assert Violation	passed
State Variable Default Visibility	passed

Reentrancy	passed
Unprotected SELFDESTRUCT Instruction	passed
Unprotected Ether Withdrawal	passed
Unchecked Call Return Value	passed
Floating Pragma	not passed
Outdated Compiler Version	passed
Integer Overflow and Underflow	passed
Function Default Visibility	passed

Classification of issue severity

High severity	High severity issues can cause a significant or full loss of funds, change of contract ownership, major interference with contract logic. Such issues require immediate attention.
Medium severity	Medium severity issues do not pose an immediate risk, but can be detrimental to the client's reputation if exploited. Medium severity issues may lead to a contract failure and can be fixed by modifying the contract state or redeployment. Such issues require attention.
Low severity	Low severity issues do not cause significant destruction to the contract's functionality. Such issues are recommended to be taken into consideration.

Issues

High severity issues

No issues were found

Medium severity issues

No issues were found

Low severity issues

1. Floating Pragma (Multiple contract)

Contracts should be deployed with the same compiler version and flags that they have been tested with thoroughly. Locking the pragma helps to ensure that contracts do not accidentally get deployed using, for example, an outdated compiler version that might introduce bugs that affect the contract system negatively.

Recommendation: Lock the pragma version and also consider known bugs (link) for the compiler version that is chosen.

2. Lack of zero-check on functions (StrategyQuartzLP)

The StratManager contract in 1464-1468L, 1482L, 1491L, 1499L, 1507L, 1518L does not validate input address values. If you fill these variables with zeros, then the contracts will no longer work correctly.

Recommendation: Add input validations with a requirement in constructor().

3. Multiplication after division (StrategyQuartzLP)

In L1860 there is a multiplication after a division, which increases the possibility of an error in integer math.

Recommendation: It is recommended to change the order of calculation of the expression in these lines.

4. Variables should be immutable (StrategyQuartzLP)

State variables in the 1592-1593L must be declared immutable. This will save gas on reading them.

Recommendation: Make these state variables immutable.

5. Call fee is not capped (StrategyQuartzLP)

In the FeeManager contract the setCallFee() function does not regulate fee changes in protocolFee variable, which if the owner is compromised, can lead to a loss of funds for the users.

Recommendation: Restrict the inputs in the setCallFee() function via require(). Check that _fee falls within a particular range of values (for example, from 5 to 20).

6. Strategy config (StrategyQuartzLP)

The pendingRewardsFunctionName variable, which has the set function on L1825, has a dependency on the function name of the MasterChef contract. If the setPendingRewardsFunctionName() function is incorrectly changed, the contract might malfunction.

Recommendation: You need to make a config for the strategy and keep it in an immutable variable. The config must include the pendingRewardsFunctionName variable. The setPendingRewardsFunctionName() function is recommended to be removed.

7. Lack of events (StrategyQuartzLP)

Many functions from the contracts lack events:

- StratManager:setKeeper()

- StratManager:setStrategist()

- StratManager:setUnirouter()

- StratManager:setVault()

- StratManager:setProtocolFeeRecipient()

- FeeManager:setCallFee()

- FeeManager:setWithdrawalFee()

- StrategyQuartzLP:setPendingRewardsFunctionName()

- StrategyQuartzLP:setHarvestOnDeposit()

- StrategyQuartzLP:setOutputToLp0()

- StrategyQuartzLP:setOutputToLp1()

Recommendation: It is recommended to create events for these functions to ensure transparency of operations.

Conclusion

Quartz Vault Multiple contract, StrategyQuartzLP contracts were audited. 7 low severity issues were found.

Users should check that they interact with the same contracts that were been audited.

Disclaimer

Slither output

```
StrategyQuartzLP.retireStrat() (contracts/strategies/quartz/
StrategyQuartzLP.sol#310-317) ignores return value by
IERC20(want).transfer(vault,wantBal) (contracts/strategies/quartz/
StrategyQuartzLP.sol#316)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unchecked-
transfer

StrategyQuartzLP.callReward() (contracts/strategies/quartz/
StrategyQuartzLP.sol#281-297) performs a multiplication on the result of a division:
        -nativeOut.mul(3).div(100).mul(callFee).div(MAX_FEE) (contracts/strategies/
quartz/StrategyQuartzLP.sol#296)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#divide-before-
multiply

QuartzVault.deposit(uint256) (contracts/QuartzVault.sol#102-128) uses a dangerous
strict equality:
        - totalSupply() == 0 (contracts/QuartzVault.sol#121)
QuartzVault.getPricePerFullShare() (contracts/QuartzVault.sol#86-89) uses a dangerous
strict equality:
        - totalSupply() == 0 (contracts/QuartzVault.sol#87-88)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dangerous-
strict-equalities

Reentrancy in QuartzVault.upgradeStrat() (contracts/QuartzVault.sol#193-211):
        External calls:
        - strategy.retireStrat() (contracts/QuartzVault.sol#205)
        State variables written after the call(s):
        - stratCandidate.implementation = address(0) (contracts/QuartzVault.sol#207)
        - stratCandidate.proposedTime = 5000000000 (contracts/QuartzVault.sol#208)
        - strategy = IStrategy(stratCandidate.implementation) (contracts/
QuartzVault.sol#206)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-
vulnerabilities-1

StrategyQuartzLP.withdraw(uint256) (contracts/strategies/quartz/
StrategyQuartzLP.sol#114-138) uses tx.origin for authorization: tx.origin != owner()
&& ! paused() (contracts/strategies/quartz/StrategyQuartzLP.sol#128)
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dangerous-
usage-of-txorigin

StrategyQuartzLP.callReward().nativeOut (contracts/strategies/quartz/
StrategyQuartzLP.sol#283) is a local variable never initialized
StrategyQuartzLP.callReward().amountOut (contracts/strategies/quartz/
StrategyQuartzLP.sol#290) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-
local-variables

StrategyQuartzLP.chargeFees(address) (contracts/strategies/quartz/
StrategyQuartzLP.sol#174-205) ignores return value by IUniswapRouterETH(unirouter).swapE
xactTokensForTokens(toNative,0,outputToNativeRoute,address(this),now) (contracts/
strategies/quartz/StrategyQuartzLP.sol#187-193)
StrategyQuartzLP.addLiquidity() (contracts/strategies/quartz/
StrategyQuartzLP.sol#208-243) ignores return value by IUniswapRouterETH(unirouter).swapE
xactTokensForTokens(outputHalf,0,outputToLp0Route,address(this),now) (contracts/
strategies/quartz/StrategyQuartzLP.sol#212-218)
StrategyQuartzLP.addLiquidity() (contracts/strategies/quartz/
StrategyQuartzLP.sol#208-243) ignores return value by IUniswapRouterETH(unirouter).swapE
xactTokensForTokens(outputHalf,0,outputToLp1Route,address(this),now) (contracts/
strategies/quartz/StrategyQuartzLP.sol#222-228)
StrategyQuartzLP.addLiquidity() (contracts/strategies/quartz/
StrategyQuartzLP.sol#208-243) ignores return value by IUniswapRouterETH(unirouter).addLi
quidity(lpToken0,lpToken1,lp0Bal,lp1Bal,1,1,address(this),now) (contracts/strategies/
quartz/StrategyQuartzLP.sol#233-242)
StrategyQuartzLP.callReward() (contracts/strategies/quartz/
StrategyQuartzLP.sol#281-297) ignores return value by
IUniswapRouterETH(unirouter).getAmountsOut(outputBal,outputToNativeRoute) (contracts/
strategies/quartz/StrategyQuartzLP.sol#285-292)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return

StratManager.setKeeper(address) (contracts/strategies/common/StratManager.sol#54-56)
should emit an event for:
        - keeper = _keeper (contracts/strategies/common/StratManager.sol#55)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-
access-control

FeeManager.setCallFee(uint256) (contracts/strategies/common/FeeManager.sol#20-25)
should emit an event for:
        - callFee = _fee (contracts/strategies/common/FeeManager.sol#23)
        - protocolFee = MAX_FEE - STRATEGIST_FEE - callFee (contracts/strategies/common/
FeeManager.sol#24)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-
arithmetic

StratManager.constructor(address,address,address,address,address)._keeper (contracts/
strategies/common/StratManager.sol#31) lacks a zero-check on :
                - keeper = _keeper (contracts/strategies/common/StratManager.sol#37)
StratManager.constructor(address,address,address,address,address)._strategist
(contracts/strategies/common/StratManager.sol#32) lacks a zero-check on :
                - strategist = _strategist (contracts/strategies/common/
StratManager.sol#38)
StratManager.constructor(address,address,address,address,address)._unirouter (contracts/
strategies/common/StratManager.sol#33) lacks a zero-check on :
                - unirouter = _unirouter (contracts/strategies/common/
StratManager.sol#39)
StratManager.constructor(address,address,address,address,address)._vault (contracts/
strategies/common/StratManager.sol#34) lacks a zero-check on :
                - vault = _vault (contracts/strategies/common/StratManager.sol#40)
StratManager.constructor(address,address,address,address,address)._protocolFeeRecipient
(contracts/strategies/common/StratManager.sol#35) lacks a zero-check on :
                - protocolFeeRecipient = _protocolFeeRecipient (contracts/strategies/
common/StratManager.sol#41)
StratManager.setKeeper(address)._keeper (contracts/strategies/common/
StratManager.sol#54) lacks a zero-check on :
                - keeper = _keeper (contracts/strategies/common/StratManager.sol#55)
StratManager.setStrategist(address)._strategist (contracts/strategies/common/
StratManager.sol#62) lacks a zero-check on :
                - strategist = _strategist (contracts/strategies/common/
StratManager.sol#64)
StratManager.setUnirouter(address)._unirouter (contracts/strategies/common/
StratManager.sol#71) lacks a zero-check on :
                - unirouter = _unirouter (contracts/strategies/common/
StratManager.sol#72)
StratManager.setVault(address)._vault (contracts/strategies/common/StratManager.sol#79)
lacks a zero-check on :
                - vault = _vault (contracts/strategies/common/StratManager.sol#80)
StratManager.setProtocolFeeRecipient(address)._protocolFeeRecipient (contracts/
strategies/common/StratManager.sol#87) lacks a zero-check on :
                - protocolFeeRecipient = _protocolFeeRecipient (contracts/strategies/
common/StratManager.sol#91)
```

```
StrategyQuartzLP.constructor(address,uint256,address,address,address,address,address,add
ress,address[],address[],address[])._want (contracts/strategies/quartz/
StrategyQuartzLP.sol#49) lacks a zero-check on :
                - want = _want (contracts/strategies/quartz/StrategyQuartzLP.sol#70)
StrategyQuartzLP.constructor(address,uint256,address,address,address,address,address,add
ress,address[],address[],address[])._chef (contracts/strategies/quartz/
StrategyQuartzLP.sol#51) lacks a zero-check on :
                - chef = _chef (contracts/strategies/quartz/StrategyQuartzLP.sol#72)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-
address-validation

Variable 'StrategyQuartzLP.callReward().amountOut (contracts/strategies/quartz/
StrategyQuartzLP.sol#290)' in StrategyQuartzLP.callReward() (contracts/strategies/
quartz/StrategyQuartzLP.sol#281-297) potentially used before declaration: nativeOut =
amountOut[amountOut.length - 1] (contracts/strategies/quartz/StrategyQuartzLP.sol#291)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#pre-
declaration-usage-of-local-variables

Reentrancy in StrategyQuartzLP.deposit() (contracts/strategies/quartz/
StrategyQuartzLP.sol#105-112):
        External calls:
        - IMasterChef(chef).deposit(poolId,wantBal) (contracts/strategies/quartz/
StrategyQuartzLP.sol#109)
        Event emitted after the call(s):
        - Deposit(balanceOf()) (contracts/strategies/quartz/StrategyQuartzLP.sol#110)
Reentrancy in StrategyQuartzLP.withdraw(uint256) (contracts/strategies/quartz/
StrategyQuartzLP.sol#114-138):
        External calls:
        - IMasterChef(chef).withdraw(poolId,_amount.sub(wantBal)) (contracts/strategies/
quartz/StrategyQuartzLP.sol#120)
        - IERC20(want).safeTransfer(vault,wantBal) (contracts/strategies/quartz/
StrategyQuartzLP.sol#135)
        Event emitted after the call(s):
        - Withdraw(balanceOf()) (contracts/strategies/quartz/StrategyQuartzLP.sol#137)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-
vulnerabilities-3

QuartzVault.getPricePerFullShare() (contracts/QuartzVault.sol#86-89) uses timestamp for
comparisons
        Dangerous comparisons:
        - totalSupply() == 0 (contracts/QuartzVault.sol#87-88)
```

```
QuartzVault.deposit(uint256) (contracts/QuartzVault.sol#102-128) uses timestamp for
comparisons
        Dangerous comparisons:
        - totalSupply() == 0 (contracts/QuartzVault.sol#121)
QuartzVault.withdraw(uint256) (contracts/QuartzVault.sol#152-168) uses timestamp for
comparisons
        Dangerous comparisons:
        - b < r (contracts/QuartzVault.sol#157)
        - _diff < _withdraw (contracts/QuartzVault.sol#162)
QuartzVault.upgradeStrat() (contracts/QuartzVault.sol#193-211) uses timestamp for
comparisons
        Dangerous comparisons:
        - require(bool,string)(stratCandidate.implementation != address(0),There is no
candidate) (contracts/QuartzVault.sol#194-197)
        - require(bool,string)(stratCandidate.proposedTime.add(approvalDelay) <
block.timestamp,Delay has not passed) (contracts/QuartzVault.sol#198-201)
QuartzVault.inCaseTokensGetStuck(address) (contracts/QuartzVault.sol#217-222) uses
timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(_token != address(want()),!token) (contracts/
QuartzVault.sol#218)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-
timestamp

FeeManager.protocolFee (contracts/strategies/common/FeeManager.sol#18) is set pre-
construction with a non-constant function or state variable:
        - MAX_FEE - STRATEGIST_FEE - callFee
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#function-
initializing-state

Pragma version^0.6.0 (contracts/QuartzVault.sol#2) allows old versions
Pragma version^0.6.0 (contracts/interfaces/common/IMasterChef.sol#3) allows old
versions
Pragma version>=0.6.0<0.9.0 (contracts/interfaces/common/IUniswapRouterETH.sol#3) is
too complex
Pragma version^0.6.0 (contracts/interfaces/common/IUniswapV2Pair.sol#2) allows old
versions
Pragma version^0.6.0 (contracts/interfaces/quartz/IStrategy.sol#2) allows old versions
Pragma version^0.6.0 (contracts/strategies/quartz/StrategyQuartzLP.sol#3) allows old
versions
Pragma version>=0.6.0<0.8.0 (contracts/utils/StringUtils.sol#3) is too complex
```

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-
versions-of-solidity

Pragma version^0.6.0 (contracts/interfaces/sushi/IMiniChefV2.sol#3) allows old versions
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-
versions-of-solidity

Pragma version^0.6.0 (contracts/interfaces/sushi/IRewarder.sol#3) allows old versions
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-
versions-of-solidity

Parameter QuartzVault.deposit(uint256)._amount (contracts/QuartzVault.sol#102) is not
in mixedCase
Parameter QuartzVault.withdraw(uint256)._shares (contracts/QuartzVault.sol#152) is not
in mixedCase
Parameter QuartzVault.proposeStrat(address)._implementation (contracts/
QuartzVault.sol#174) is not in mixedCase
Parameter QuartzVault.inCaseTokensGetStuck(address)._token (contracts/
QuartzVault.sol#217) is not in mixedCase
Parameter FeeManager.setCallFee(uint256)._fee (contracts/strategies/common/
FeeManager.sol#20) is not in mixedCase
Parameter FeeManager.setWithdrawalFee(uint256)._fee (contracts/strategies/common/
FeeManager.sol#27) is not in mixedCase
Parameter StratManager.setKeeper(address)._keeper (contracts/strategies/common/
StratManager.sol#54) is not in mixedCase
Parameter StratManager.setStrategist(address)._strategist (contracts/strategies/common/
StratManager.sol#62) is not in mixedCase
Parameter StratManager.setUnirouter(address)._unirouter (contracts/strategies/common/
StratManager.sol#71) is not in mixedCase
Parameter StratManager.setVault(address)._vault (contracts/strategies/common/
StratManager.sol#79) is not in mixedCase
Parameter StratManager.setProtocolFeeRecipient(address)._protocolFeeRecipient
(contracts/strategies/common/StratManager.sol#87) is not in mixedCase
Parameter StrategyQuartzLP.withdraw(uint256)._amount (contracts/strategies/quartz/
StrategyQuartzLP.sol#114) is not in mixedCase
Parameter
StrategyQuartzLP.setPendingRewardsFunctionName(string)._pendingRewardsFunctionName
(contracts/strategies/quartz/StrategyQuartzLP.sol#262) is not in mixedCase
Parameter StrategyQuartzLP.setHarvestOnDeposit(bool)._harvestOnDeposit (contracts/
strategies/quartz/StrategyQuartzLP.sol#299) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-
```

```
solidity-naming-conventions

Variable IUniswapRouterETH.addLiquidity(address,address,uint256,uint256,uint256,uint256,
address,uint256).amountADesired (contracts/interfaces/common/IUniswapRouterETH.sol#9)
is too similar to IUniswapRouterETH.addLiquidity(address,address,uint256,uint256,uint256
,uint256,address,uint256).amountBDesired (contracts/interfaces/common/
IUniswapRouterETH.sol#10)
Variable StrategyQuartzLP.constructor(address,uint256,address,address,address,address,ad
dress,address,address[],address[],address[])._outputToLp0Route (contracts/strategies/
quartz/StrategyQuartzLP.sol#58) is too similar to StrategyQuartzLP.constructor(address,u
int256,address,address,address,address,address,address,address[],address[],address[])._o
utputToLp1Route (contracts/strategies/quartz/StrategyQuartzLP.sol#59)
Variable StrategyQuartzLP.outputToLp0Route (contracts/strategies/quartz/
StrategyQuartzLP.sol#37) is too similar to StrategyQuartzLP.outputToLp1Route (contracts/
strategies/quartz/StrategyQuartzLP.sol#38)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-
are-too-similar

QuartzVault.upgradeStrat() (contracts/QuartzVault.sol#193-211) uses literals with too
many digits:
        - stratCandidate.proposedTime = 5000000000 (contracts/QuartzVault.sol#208)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-
digits

getPricePerFullShare() should be declared external:
        - QuartzVault.getPricePerFullShare() (contracts/QuartzVault.sol#86-89)
proposeStrat(address) should be declared external:
        - QuartzVault.proposeStrat(address) (contracts/QuartzVault.sol#174-185)
upgradeStrat() should be declared external:
        - QuartzVault.upgradeStrat() (contracts/QuartzVault.sol#193-211)
setCallFee(uint256) should be declared external:
        - FeeManager.setCallFee(uint256) (contracts/strategies/common/
FeeManager.sol#20-25)
callReward() should be declared external:
        - StrategyQuartzLP.callReward() (contracts/strategies/quartz/
StrategyQuartzLP.sol#281-297)
panic() should be declared external:
        - StrategyQuartzLP.panic() (contracts/strategies/quartz/
StrategyQuartzLP.sol#320-323)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-
function-that-could-be-declared-external
```

```
. analyzed (20 contracts with 77 detectors), 69 result(s) found
```

Guard