QVAULT

Security Assessment and Code Audit

Mundus

29 Sept 2025

INTRODUCTION

The QVAULT Smart Contract’s changes focus on updating several different types of proposals, each with its own cost and voting requirements. Below are how they work

- QCAP general proposal: electing team members, no impact on QCAP’s investment
- Quorum requirement: adjustments to the quorum requirement
- IPO Participation: Investors cast votes to determine their investment amount in an IPO. QVAULT computes a weighted average, proportional to the number of $QCAP tokens held by each investor. Proposals are limited by the total Qubic amount available within the smart contract.
- QEarn participation: after a proposal gets approval, QVAULT locks a capital amount for a set period. Upon return, only the earned interest is distributed as revenue; the original capital remains in the QVAULT.
- Fundraising Proposal: If a proposal, such as the sale of assets, is approved, QVAULT will facilitate the sale of those tokens. The Qubic raised from this type of proposal will not be considered revenue and, therefore, should not be distributed.
- Marketplace proposal: a proposal, such as the sale of assets, is approved, the user receives payment. If not, or if QVAULT lacks sufficient funds, the assets are returned to the user.
    - An example selling assets: 10 Quottery and 1 Qx for 10B $Qubic and 10,000 $QCAP
- Allocation percentage: a proposal can change revenue allocation percentages, but the QVAULT revenue share is permanently fixed at 3%. The team's 2% fee will be zeroed out on January 1, 2027, and those funds will be reallocated to revenue distribution.
- Fund management: IPOs have the highest priority, followed by the marketplace, and then QEarn.

AUDIT PROCESS

Following considerations have been tested and reviewed by Mundus team:

- Reviewing google test files that are written by SC developer.
- Trying several attack vectors to exploit the SC.
- Ensuring contract logic meets the specifications.
- Thorough line-by-line manual review.

RESULTS

We identified a range of security issues in our assessment, from critical to minor. We recommend fixing them to meet security standards and best practices.

OVERVIEW

1. Project name: **QVAULT**
2. Platform: **QUBIC SC system**
3. Is new SC deployment: **NO**
4. Is upgraded SC: **YES**
5. Project size: Large
6. List of github commits to review: https://github.com/qubic/core/pull/365/commits

RESULTS SUMMARY

Total Issues	30
Critical	0
Major	3
Medium	15
Minor	12

AUDIT SCOPE

Filename	SHA	Latest commit
QVault.h	a73ed4f5cf268d770798ea50a17733d41503a378	fix: update for the rev2 audit
contract_qvault.cpp (gtest)	a73ed4f5cf268d770798ea50a17733d41503a378	fix: update for the rev2 audit

SUMMARY OF FINDINGS

ID	Title	Category	Severity	Status
QVAULT-01	Numerous problems were identified in the current voteInProposal procedure's implementation	Logic	Major	Fixed
QVAULT-02	Centralization issue	Gov	Major	Fixed
QVAULT-03	Not expandable size when QUBIC SC expanding	Scalable	Medium	Fixed
QVAULT-04	The API call needs a check to see if it failed or succeeded	Logic	Medium	Fixed
QVAULT-05	Not verify that the number of stakers is within the accepted range	Logic	Medium	Fixed
QVAULT-06	Verify input amount for unStake procedure	Logic	Medium	Fixed
QVAULT-07	Invocators with QVAULT shares can vote regardless of voting power	Logic	Medium	Fixed
QVAULT-08	Incorrect updates if not verify input proposalType	Logic	Medium	Fixed
QVAULT-09	Lack of a return code and unvalidated input proposalId	Code convention	Medium	Fixed
QVAULT-10	Found several issues with getIdentitiesHvVtPw	Logic	Medium	Fixed

QVAULT-11	Found two issues with getQcapBurntAmountInLastEpoches	Logic	Medium	Fixed
QVAULT-12	The temporary counter was not initialized	Logic	Medium	Fixed
QVAULT-13	Found several issues with getAmountForQearnInUpcomingEpoch	Logic	Medium	Fixed
QVAULT-14	Avoid hardcoded numbers	Code convention	Minor	Fixed
QVAULT-15	Make a definition MAX_URLS_COUNT	Code convention	Minor	Fixed
QVAULT-16	Make a definition MIN_VOTING_POWER	Code convention	Minor	Fixed
QVAULT-17	Make definitions for proposal results	Code convention	Minor	Fixed
QVAULT-18	Limit proposals not verified	Logic	Minor	Fixed
QVAULT-19	Port utility functions into QVAULT	Code convention	Minor	Fixed
QVAULT-20	Make a definition for allocation percentages	Code convention	Minor	Fixed
QVAULT-21	Make definition for maximum of countOfVote	Code convention	Minor	Fixed
QVAULT-22	Lack of a returnCode for getStakedAmountAndVotingPower	Code convention	Minor	Fixed

QVAULT-23	Lack of a returnCode for ppCreationPower	Code convention	Minor	Fixed
QVAULT-24	Verify the amount before transferring	Logic	Minor	Fixed
QVAULT-25	Lack of asset handling mechanism for transferred assets when the number of stakers hits the limit	Logic	Major	Fixed
QVAULT-26	The API call needs a check to see if it failed or succeeded	Logic	Medium	Fixed
QVAULT-27	Port utility functions into Qvault	Code convention	Minor	Fixed
QVAULT-28	Need to verify proposalId with number-of-proposals of input proposal's type	Logic	Medium	Fixed
QVAULT-29	Need to verify input proposalId in multiple procedures/functions	Logic	Medium	Fixed
QVAULT-30	Need to re-exam returnCode in ppCreationPower	Logic	Medium	Fixed

FINDINGS

QVAULT-01 Numerous problems were identified in the current voteInProposal procedure's implementation

- ❖ Latest commit: f63a6807
- ❖ File(s) affected: QVault.h:1508-1673
- ❖ Description: Numerous problems were identified in the current voteInProposal procedure's implementation.
    - ➢ The proposalId is a 0-based index. Therefore, the check should be input.proposalId >= QVAULT_MAX_NUMBER_OF_PROPOSAL.
    - ➢ input.proposalId: based on input.proposalType, retrieve the total number of proposals. Verify that input.proposalId is within the valid range (i.e., less than the total number of proposals). Failure to do so may lead to unexpected results in subsequent calls to get(input.proposalId).
    - ➢ The existing case for locals._r has not been initialized, which could lead to nondeterministic checks at lines 1526 or 1531.
    - ➢ locals.countOfVote and locals.newVoteList: if lines 1515 or 1417 return false, unexpected results will occur because they have not been initialized.
    - ➢ Signed/unsigned mismatch at some += operations
        - ■ numberOfYes += locals.numberOfYes,
        - ■ numberOfNo += locals.numberOfNo,
        - ■ totalWeight +=
    - ➢ Lines 1583 & 1601 & 1667: To enhance code clarity, need to add more comments indicating that an invocator can only vote for a single proposal.
- ❖ Recommendation:

QVAULT-02 Centralization issue

- Latest commit: f63a6807
- File(s) affected: QVault.h
- Description: **adminAddress**, is hardcoded, cannot be changed by shareholders.
- Recommendation:

QVAULT-03 Not expandable size when QUBIC SC expanding

- Latest commit: f63a6807
- File(s) affected: QVault.h:383, 384, 555, 556
- Description: Hardcoded, not expandable size when QUBIC SC expanding
- Recommendation: Make a definition for `1048576` then multiply X_MULTIPLIER.

QVAULT-04 The API call needs a check to see if it failed or succeeded

- Latest commit: f63a6807
- File(s) affected: QVault.h:696, 754, 1762
- Description: The API call needs a check to see if it failed or succeeded.
- Recommendation:

QVAULT-05 Not verify that the number of stakers is within the accepted range.

- Latest commit: f63a6807
- File(s) affected: QVault.h:710
- Description: Check for limitations before increasing the numberOfStaker.
- Recommendation:

QVAULT-06 Verify input amount for unStake procedure

- Latest commit: f63a6807
- File(s) affected: QVault.h:740
- Description: Need to check input.amount > 0
- Recommendation:

QVAULT-07 Invocators with QVAULT shares can vote regardless of voting power

- Latest commit: f63a6807
- File(s) affected: QVault.h:800, 884, 980, 1088, 1185, 1388
- Description: An invocator can still vote or submit a proposal, even without sufficient voting power, if they possess QVAULT share(s). Therefore, it is necessary to first check the numberOfShares() at pointed positions above to avoid missing cases.
- Recommendation:

QVAULT-08 Incorrect updates if not verify input proposalType

- Latest commit: f63a6807
- File(s) affected: QVault.h:1509
- Description: At the beginning of the procedure, input.proposalType must be validated to ensure it falls within the valid range. Failure to do so will result in incorrect updates to state.vote and state.countOfVote when combined with randomized result of locals.statusOfProposal (It was observed that locals.statusOfProposal lacks initialization).
- Recommendation:

QVAULT-09 Lack of a return code and unvalidated input proposalId

- Latest commit: f63a6807
- File(s) affected: QVault.h:1920, 1949, 1978, 2007, 2036, 2065, 2094
- Description: Requires to verify input proposalId falls within the acceptable range (number of proposals) and add a returnCode.
- Recommendation:

QVAULT-10 Found several issues with getIdentitiesHvVtPw

- Latest commit: f63a6807
- File(s) affected: QVault.h:2135-2150
- Description: getIdentitiesHvVtPw presented several issues
    - lacks of a returnCode for output
    - not check input.count is valid
    - not reset/empty output (idList & amountList) at begin of function
- Recommendation:

QVAULT-11 Found two issues with getQcapBurntAmountInLastEpoches

- Latest commit: f63a6807
- File(s) affected: QVault.h:2250-2260
- Description: getQcapBurntAmountInLastEpoches presented two issues
    - lacks of a returnCode for output
    - not reset/empty/initialize output.burntAmount
- Recommendation:

QVAULT-12 The temporary counter was not initialized

- Latest commit: f63a6807
- File(s) affected: QVault.h:2491, 2504
- Description: The variable locals.count is used before it has been initialized.
- Recommendation:

QVAULT-13 Found several issues with getAmountForQearnInUpcomingEpoch

- Latest commit: f63a6807
- File(s) affected: QVault.h:2544-2554
- Description: getAmountForQearnInUpcomingEpoch presented two issues
  - lacks of a returnCode for output
  - not reset/empty/initialize output.amount
- Recommendation:

QVAULT-14 Avoid hardcoded numbers

- Latest commit: f63a6807
- File(s) affected: QVault.h:651, 652
- Description: Make definitions for minQuorumRq/maxQuorumRq instead of hardcoded numbers
- Recommendation:

QVAULT-15 Make a definition MAX_URLS_COUNT

- Latest commit: f63a6807
- File(s) affected: QVault.h:153, 173, ...
- Description: Make a definition MAX_URLS_COUNT and use it instead of hardcoded 256 everywhere
- Recommendation:

QVAULT-16 Make a definition MIN_VOTING_POWER

- Latest commit: f63a6807
- File(s) affected: QVault.h:795. 879. 975. 1083. 1179. 1382. 2192
- Description: Make a definition MIN_VOTING_POWER then using it instead of hardcoded 10000 everywhere
- Recommendation:

QVAULT-17 Make definitions for proposal results

- Latest commit: f63a6807
- File(s) affected: QVault.h:662, 849, 933, 1029, 1133, 1272, 1337, 1467, 1745, 2550, 2635, 2647, 2660, 2811, 2815, 2819, 2842, 2846, 2850, 2870, 2892. 2896, 2900, 2964, 2968, 2972, 3024, 3028, 3032, 3033, 3042, ...
- Description: Make definitions for proposal results instead of hard-coded values. Currently, the proposal result is hardcoded as: 0=passed, 1=rejected, 2=insufficient quorum, 3=insufficient funds, 4=insufficient QCAP.
- Recommendation: Consider adding another number for the new proposal's result. Currently, it's 4, this number is labeled as insufficient QCAP.

QVAULT-18 Limit proposals not verified

- Latest commit: f63a6807
- File(s) affected: QVault.h:851, 937, 1035, 1143, 1278, 1345, 1474
- Description: Check to maximum number of proposals before add
- Recommendation: Add a check at the begin of procedure

QVAULT-19 Port utility functions into Qvault

- ❖ Latest commit: f63a6807
- ❖ File(s) affected: QVault.h:1208, 1209, 1411, 1412, 1704, 1705, 3034, 3035, 3084, 3085
- ❖ Description: Port these utility functions into Qvault, so that if they are removed or changed from outside then Qvault's logic will not be impacted.
- ❖ Recommendation:

QVAULT-20 Make a definition for allocation percentages

- ❖ Latest commit: f63a6807
- ❖ File(s) affected: QVault.h:1432
- ❖ Description: Make a definition instead of hardcoded number 970; and also add comment `Validates allocation percentages sum to 970 (per mille)`.
- ❖ Recommendation:

QVAULT-21 Make definition for maximum of countOfVote

- ❖ Latest commit: f63a6807
- ❖ File(s) affected: QVault.h:555, 1526
- ❖ Description: Make definition for maximum of countOfVote = 16
- ❖ Recommendation:

QVAULT-22 Lack of a returnCode for getStakedAmountAndVotingPower

- ❖ Latest commit: f63a6807
- ❖ File(s) affected: QVault.h:1874
- ❖ Description: Lack of a returnCode for getStakedAmountAndVotingPower
- ❖ Recommendation:

QVAULT-23 Lack of a returnCode for ppCreationPower

- ❖ Latest commit: f63a6807
- ❖ File(s) affected: QVault.h:2186-2214
- ❖ Description: Lack of a returnCode for ppCreationPower
  - ➢ not found any staker address
  - ➢ not enough power voting
  - ➢ not enough number of shares
- ❖ Recommendation:

QVAULT-24 Verify the amount before transferring

- ❖ Latest commit: f63a6807
- ❖ File(s) affected: QVault.h:2789, 2797
- ❖ Description: Verify the amount > 0 before calling qpi.transfer
- ❖ Recommendation:

QVAULT-25 Lack of asset handling mechanism for transferred assets when the number of stakers hits the limit

- ❖ Latest commit: 4de38c79
- ❖ File(s) affected: QVault.h:732-736
- ❖ Description: For new stakers, if the number of stakers hits the QVAULT_X_MULTIPLIER limit, the assets transferred on line #716 must be rolled back.
- ❖ Recommendation:

QVAULT-26 The API call needs a check to see if it failed or succeeded

- ❖ Latest commit: 4de38c79
- ❖ File(s) affected: QVault.h:789
- ❖ Description: The API call needs a check to see if it failed or succeeded.
- ❖ Recommendation:

QVAULT-27 Port utility functions into Qvault

- Latest commit: 4de38c79
- File(s) affected: QVault.h:1300, 1301, 1525, 1526, 1832, 1833, 3237, 3238, 3287, 3288
- Description: Port these utility functions into Qvault, so that if they are removed or changed from outside then Qvault's logic will not be impacted.
- Recommendation:

QVAULT-28 Need to verify proposalId with number-of-proposals of input proposal's type

- Latest commit: 4de38c79
- File(s) affected: QVault.h:1638
- Description: Based on input.proposalType, retrieve the total number of proposals. Verify that input.proposalId is within the valid range (i.e., less than the total number of proposals). Failure to do so may lead to unexpected results in subsequent calls to get(input.proposalId).
- Recommendation:

QVAULT-29 Need to verify input proposalId in multiple procedures/functions

- Latest commit: 4de38c79
- File(s) affected: QVault.h:2050, 2086, 2122, 2158, 2194, 2230, 2266
- Description: Verify that input.proposalId is within the valid range (i.e., less than the current number-of-proposals **numberOfGP/numberOfQCP/numberOfIPOP…**). Failure to do so may lead to unexpected results in subsequent calls to get(input.proposalId).
- Recommendation:

QVAULT-30 Need to re-exam returnCode in ppCreationPower

- Latest commit: 4de38c79
- File(s) affected: QVault.h:2394,2401
- Description: Issues updating returnCode for the ppCreationPower function
  - Line 2394: Potential returnCode overwrite of line 2390
  - Line 2401: returnCode should be QVAULT_SUCCESS.
- Recommendation: