QuillAudits

Audit Report
February, 2024

For

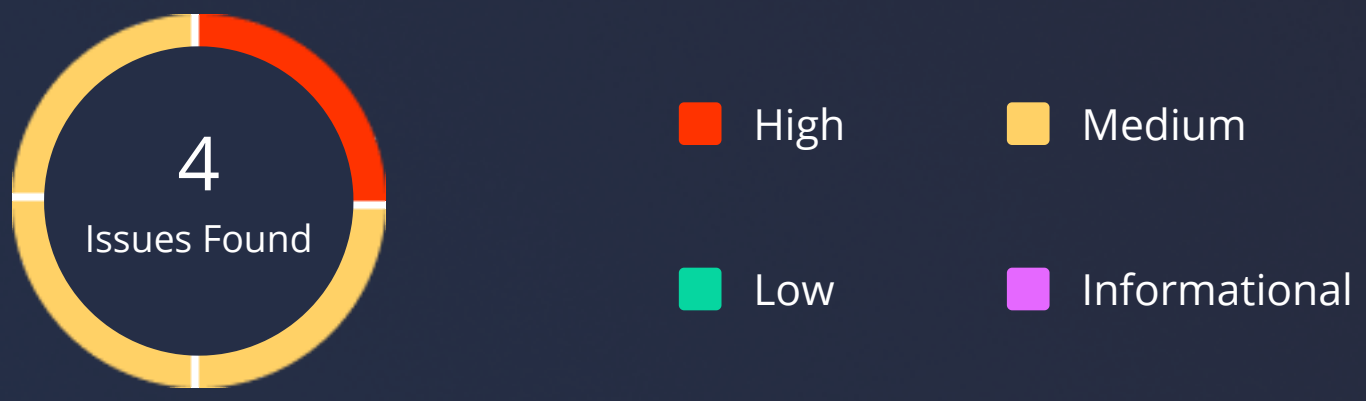
BPROXY

Table of Content

Executive Summary

Project Name	Alpha
Overview	AlphaBTC is a DeFi protocol to facilitate the exchange of BTCpx tokens for aBTCpx tokens. Users can deposit BTCpx into the contract to receive aBTCpx tokens, which can be withdrawn after a vesting period. The contract periodically distributes dividends to token holders based on the total supply of aBTCpx, the APR, and the time since the last distribution. Controlled by an owner, the contract ensures safety measures such as preventing transfers of aBTCpx tokens and recovering excess tokens. Overall, AlphaBTC offers a mechanism for users to earn rewards by depositing and holding BTCpx tokens in a transparent and trustless manner.
Timeline	12th January 2024 - 5th February 2024
Updated Code Received	6th February 2024
Second Review	8th February 2024 - 13th February 2024
Method	Manual Review, Functional Testing, Automated Testing, etc. All the raised flags were manually reviewed and re-tested to identify any false positives.
Audit Scope	The scope of this audit was to analyze the Alpha codebase for quality, security, and correctness.
Source Code	*https://github.com/Proxy-Protocol/AlphaBTC*
Branch	ddaa7d2191d185f35c9ca0f67e82dc069464f805
Fixed In	ce8c42a00667d1c0da0a9b267025fe04bee7eefd

Number of Security Issues per Severity



	High	Medium	Low	Informational
Open Issues	0	0	0	0
Acknowledged Issues	0	3	0	0
Partially Resolved Issues	0	0	0	0
Resolved Issues	1	0	0	0

Checked Vulnerabilities

- 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 contracts, care was taken to ensure:

- The overall quality of code.
- Use of best practices.
- Code documentation and comments match logic and expected behavior.
- Token distribution and calculations are as per the intended behavior 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.

Static Analysis

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

Code Review / Manual Analysis

Manual Analysis or review of code was done to identify new vulnerabilities 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 the automated analysis were manually verified.

Gas Consumption

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

Tools and Platforms used for Audit

Hardhat, Foundry.

Types of Severity

Every issue in this report has been assigned to 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 be fixed before moving to a live environment.

Medium 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 Severity Issues

Low-level severity issues can cause minor impact and 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 four severity issues that indicate an improvement request, a general question, a cosmetic or documentation error, or a request for information. There is low-to-no impact.

Types of Issues

Open

Security vulnerabilities identified that must be resolved and are currently unresolved.

Resolved

These are the issues identified in the initial audit and have been successfully fixed.

Acknowledged

Vulnerabilities which have been acknowledged but are yet to be resolved.

Partially Resolved

Considerable efforts have been invested to reduce the risk/impact of the security issue, but are not completely resolved.

High Severity Issues

1. Wrong accounting of user balance in scaledBalanceOf function

Path

https://github.com/Proxy-Protocol/AlphaBTC/blob/ddaa7d2191d185f35c9ca0f67e82dc069464f805/contracts/AlphaBTC.sol#L271-L282

Function

scaledBalanceOf()

Description

When a user deposits funds, if multiple dividend periods pass without any activity from that user, subsequent calls to the deposit or withdraw functions will calculate the user's rewards and balance using the **user's index + 1** in the **scaledBalanceOf** function. However, in the updateUser function, the **user's index is updated to the latest index**.

Proof of Concept (Add POC if Applicable)

Add this test in the test.js:

https://gist.github.com/Makg2/2404e6a3face82bfa37904ba151b39fd

Note: You may need to declare the **_liquidityIndex** variable public for this test to run.

Recommendation

Change the user's balance according to **_index** value instead of **userInfo.index**.

Status

Resolved

Medium Severity Issues

1. Updating the timestamp for previous investments will delay the user from withdrawing funds

Path

https://github.com/Proxy-Protocol/AlphaBTC/blob/ddaa7d2191d185f35c9ca0f67e82dc069464f805/contracts/AlphaBTC.sol#L413

Function

deposit()

Description

Each time the user makes a deposit, their timestamp is updated to reflect the latest block time. This means that the user will need to wait for the vesting period from their most recent deposit to withdraw any funds, including the initial deposit.

Proof of Concept (Add POC if Applicable)

Add this test in the test.js:

https://gist.github.com/Makg2/1f991e1bf666a2f06a474d00ce500344

Recommendation

Maintain different timestamps for every deposit.

Status

Acknowledged

2. If the dividend is not called by the owner then the user will not be able to receive any reward

Path

https://github.com/Proxy-Protocol/AlphaBTC/blob/ddaa7d2191d185f35c9ca0f67e82dc069464f805/contracts/AlphaBTC.sol#L589

Function

updateUser(), scaledBalanceOf()

Description

If a user deposits at some point and the owner does not change dividend during the user's deposit and withdrawal then the user will not receive any reward, no matter how much time the user has spent.

Proof of Concept (Add POC if Applicable)

Add this test in the test.js:

https://gist.github.com/Makg2/a92ab59e8727944a6fe6f4afb55bd6c7

Recommendation

Change implementation to distribute rewards to users acc. to time passed irrespective of dividend change.

Status

Acknowledged

3. Loss of user funds

Path

https://github.com/Proxy-Protocol/AlphaBTC/blob/ddaa7d2191d185f35c9ca0f67e82dc069464f805/contracts/AlphaBTC.sol#L613

Function

safeBTCpxTransfer()

Description

The function transfers funds based on the amount passed or the contract's balance, whichever is lower. However, in the withdraw function, it is utilized to transfer BTCpx tokens without retaining the user's remaining balance if a lower amount is transferred. This could result in the loss of user funds.

Recommendation

Store remaining user balance.

Status

Acknowledged

Automated Tests

No major issues were found. Some false positive errors were reported by the tools. All the other issues have been categorized above according to their level of severity.

Closing Summary

In this report, we have considered the security of the Alpha codebase. We performed our audit according to the procedure described above.

Some issues of High and medium severity were found, Some suggestions and best practices are also provided in order to improve the code quality and security posture.

Disclaimer

QuillAudits Smart contract security audit provides services to help identify and mitigate potential security risks in BTC Proxy Alpha smart contracts. However, it is important to understand that no security audit can guarantee complete protection against all possible security threats. QuillAudits audit reports are based on the information provided to us at the time of the audit, and we cannot guarantee the accuracy or completeness of this information. Additionally, the security landscape is constantly evolving, and new security threats may emerge after the audit has been completed.

Therefore, it is recommended that multiple audits and bug bounty programs be conducted to ensure the ongoing security of BTC Proxy Alpha smart contracts. One audit is not enough to guarantee complete protection against all possible security threats. It is important to implement proper risk management strategies and stay vigilant in monitoring your smart contracts for potential security risks.

QuillAudits cannot be held liable for any security breaches or losses that may occur subsequent to and despite using our audit services. It is the responsibility of the BTC Proxy Alpha team to implement the recommendations provided in our audit reports and to take appropriate steps to mitigate potential security risks.

About QuillAudits

QuillAudits is a secure smart contracts audit platform designed by QuillHash Technologies. We are a team of dedicated blockchain security experts and smart contract auditors determined to ensure that Smart Contract-based Web3 projects can avail the latest and best security solutions to operate in a trustworthy and risk-free ecosystem.

850+
Audits Completed

$30B
Secured

$30B
Lines of Code Audited

Follow Our Journey

Audit Report
February, 2024

For

PROXY

QuillAudits

Canada, India, Singapore, UAE, UK

www.quillaudits.com

audits@quillhash.com