# Security Audit Report

# BTC Plus

SECBIT

Apr 28, 2021

# 1. Introduction

BTC Plus is a positively rebasing ERC20 BTC Token, which can maintain its peg to BTC and provide global interest to all token holders. SECBIT Labs conducted an audit from Apr 1st to Apr 28th, 2021, including an analysis of Smart Contracts in 3 areas: **code bugs**, **logic flaws**, and **risk assessment**. The audit results show that BTC Plus has no critical security risks. The SECBIT team has some tips on logical implementation, potential risks, and code revising (see part 4 for details).

# 2. Project Information

This part describes the necessary information and code structure.

## 2.1 Basic information

The basic information about BTC Plus is shown below:

- Project website
    - https://acbtc.fi
- Smart contract code
    - https://github.com/nutsfinance/BTC-Plus, commit e9e0272.

## 2.2 Contract List

The following content shows the main contracts included in BTC Plus project:

| Contract | Description |
|---|---|
| CompositePlus.sol | Composite plus token. |
| Migrations.sol | Migration contract. |
| Plus.sol | Plus token base contract. |
| SinglePlus.sol | Single plus token. |
| composite/BTC+.sol | BTC+ token contract. |
| governance/GaugeController.sol | Controller for all liquidity gauges. |
| governance/LiquidityGauge.sol | Liquidity gauge that stakes token and earns |

| | |
|---|---|
| | reward. |
| governance/Timelock.sol | Time lock to delay transaction executions. |
| misc/AutoBTC.sol | Tokenization of AutoFarm's BTCB position. |
| misc/AutoBTCv2.sol | Tokenization V2 of AutoFarm's BTCB position. |
| misc/BTCZapBsc.sol | Zap for BTC plus on BSC. |
| misc/Claimer.sol | A utility contract that helps to claims from multiple liquidity gauges. |
| misc/ERC20Proxy.sol | Proxy for ERC20 tokens. |
| misc/GaugeControllerProxy.sol | Proxy for gauge controller. |
| misc/LiquidityGaugeProxy.sol | Proxy for liquidity gauge. |
| misc/UpdatableLiquidityGauge.sol | Liquidity gauge that can update. |
| misc/VotingEscrowProxy.sol | Proxy for Voting Escrow. |
| single/bsc/ACoconutBTC-BSC+.sol | Single plus for ACoconutBTC-BSC+. |
| single/bsc/ACryptoSBTC+.sol | Single plus for ACryptoS BTC. |
| single/bsc/AutoBTC+.sol | Single Plus for AutoFarm BTC+. |
| single/bsc/AutoBTCv2+.sol | Single Plus for AutoFarm BTC v2+. |
| single/bsc/ForTubeBTCB+.sol | Single Plus for ForTube BCTB. |
| single/bsc/VenusBTC+.sol | Single plus of Venus BTC. |
| single/eth/AaveWBTC+.sol | Single Plus for Aave v2 WBTC. |

| | |
|---|---|
| single/eth/ACoconutBTC+.sol | Single plus for ACoconut BTC+. |
| single/eth/CompoundWBTC+.sol | Single plus of Compound WBTC. |
| single/eth/RenCrv+.sol | Single plus for renCrv. |
| single/eth/SbtcCrv+.sol | Single plus for sbtcCrv. |
| single/eth/VesperWBTC+.sol | Single plus for Vesper WBTC. |

# 3. Code Analysis

This part describes code assessment details, including two items: "role classification" and "functional analysis".

## 3.1 Role Classification

There are several key roles in BTC Plus, namely Governance Account, Strategist, Claimer, and Common Account.

- Governance Account
    - Description The account performing governance
    - Authority
        - Update governance account
        - Update contract parameters
        - Add/remove rebalancers to/from rebalancer list
        - Add/remove rebase hooks to/from transaction list
        - Add/remove plus tokens to/from basket
        - Add/remove liquidity gauge to/from the gauge controller
        - Salvage any ETH/Token deposited to gauge contract by mistake
        - All authorities of strategist
    - Method of Authorization The creator of the contract, or authorized by the transferring of governance account
- Strategist
    - Description The account performing strategies
    - Authority
        - Update strategist
        - Update the mint paused state of a token
        - Remove an existing rebalancer
        - Rebalance the basket

        - Perform actions for yield and investment (invest, divest, harvest)
        - Salvage any ETH/Token deposited to plus tokens
    - Method of Authorization Authorized by governance account or strategist
- Claimer
    - Description The account claiming rewards
    - Authority
        - Claim reward on behalf of users in gauges
        - Kick an account for abusing the boosting in gauges
    - Method of Authorization Authorized by governance account
- Common Account
    - Description The accounts holding supported BTC tokens
    - Authority
        - All authorities of ERC20 Token
        - Mint the plus token with the underlying tokens
        - Redeem the plus tokens for underlying tokens
        - Stake tokens to earn rewards in liquidity gauges
    - Method of Authorization Holders of the supported token

## 3.2 Functional Analysis

BTC Plus is a positively rebasing ERC20 BTC Token. We can divide the critical functions of the BTC Plus contract into several parts:

### Plus Token

Users can mint Plus Token with both ERC20 BTC token or ERC20 BTC LP token. The Plus Token balance increases as interest are accrued with the tokens used to mint Plus Token.

There are two types of Plus Tokens, namely Composite Plus Token (BTC+) and Single Plus Token (renCrv+, cWBTC+, etc.). Single Plus Token is pegged by BTC ERC20 Tokens, and a basket of Single Plus Tokens backs composite Plus Token.

The main functions in Plus Token are as below:

- `mint()` for minting plus tokens with underlying tokens
- `redeem()` for redeeming plus tokens for underlying tokens
- `rebase()` for accruing interest to increase index
- `rebalance()` for rebalancing the basket for a better yield
- `invest()` for investing the underlying assets for additional yield
- `harvest()` for additional harvesting yield from the investment
- `divest()` for retrieving the underlying assets from the investment

### Liquidity Gauge

Users can stake Plus Tokens in liquidity gauges to earn rewards, which provides an additional yield for Plus Tokens.

The main functions in Liquidity Gauge are as below:

- `deposit()` for staking underlying Plus Token
- `withdraw()` for withdrawing the staked token from liquidity gauge.

# 4. Audit Detail

This part describes the process and detailed results of the audit and demonstrates the problems and potential risks.

## 4.1 Audit Process

The audit strictly followed the audit specification of SECBIT Labs. We analyzed the project from code bug, logical implementation, and potential risks. The process consists of four steps:

- Fully analysis of code line by line.
- Evaluation of vulnerabilities and potential risks revealed in the source code.
- Communication on assessment and confirmation.
- Audit report writing.

### 4.2 Audit Result

After scanning with adelaide, sf-checker, and badmsg.sender (internal version) developed by SECBIT Labs and open source tools including Mythril, Slither, SmartCheck, and Securify, the auditing team performed a manual assessment. The team inspected the contract line by line, and the result could be categorized into twenty-one types:

| Number | Classification | Result |
|---|---|---|
| 1 | Normal functioning of features defined by the contract | ✓ |
| 2 | No obvious bug (e.g., overflow, underflow) | ✓ |
| 3 | Pass Solidity compiler check with no potential error | ✓ |
| 4 | Pass common tools check with no obvious vulnerability | ✓ |
| 5 | No obvious gas-consuming operation | ✓ |
| 6 | Meet with ERC20 | ✓ |
| 7 | No risk in low-level call (call, delegatecall, callcode) and in-line assembly | ✓ |
| 8 | No deprecated or outdated usage | ✓ |
| 9 | Explicit implementation, visibility, variable type, and Solidity version number | ✓ |

| | | |
|---|---|---|
| 10 | No redundant code | ✓ |
| 11 | No potential risk manipulated by timestamp and network environment | ✓ |
| 12 | Explicit business logic | ✓ |
| 13 | Implementation consistent with annotation and other info | ✓ |
| 14 | No hidden code about any logic that is not mentioned in design | ✓ |
| 15 | No ambiguous logic | ✓ |
| 16 | No risk threatening the developing team | ✓ |
| 17 | No risk threatening exchanges, wallets, and DApps | ✓ |
| 18 | No risk threatening token holders | ✓ |
| 19 | No privilege on managing others' balances | ✓ |
| 20 | No unnecessary minting method | ✓ |
| 21 | Correct managing hierarchy | ✓ |

## 4.3 Risks

### 4.3.1 Compatibility with other applications

| Risk Type | Risk Level | Impact | Status |
|---|---|---|---|
| Design & Implementation | Low | Compatibility | Discussed |

**Description**

The Plus Token is a modified ERC20 implementation, and there may be some compatibility issues when integrating with other applications. For example, the user's balance is designed and implemented to grow automatically as the system rebases. As another example, transferring tokens currently do not emit any `transfer` events, making it difficult to track the flow of funds and the distribution of holders on blockchain explorers like Etherscan.

**Suggestion**

Carefully consider the accessibility to other applications. Think about adding `transfer` events for the token transfer function.

**Status**

The risk has been discussed.

### 4.3.2 Potential unknown risks from underlying assets and external protocols

| Risk Type | Risk Level | Impact | Status |
|---|---|---|---|
| External Risks | Medium | Single Point Failure | Discussed |

### Description

Each Single Plus Token connects to a different DeFi protocol. The Composite Plus Token combines these tokens into the BTC+ Token. However, due to the complexity and composability of DeFi, different DeFi protocols may carry unknown risks. In addition to this, some underlying assets or farming strategies use leverage to maximize returns. Protocol maintainers need to build a robust system to maintain a healthy collateral rate for preventing liquidation.

### Suggestion

It is recommended that the Composite Plus Token only adds Single Plus Token that is less risky and has been running smoothly for a long time. The BTC Plus protocol should attempt to risk-grade the different protocols and farming strategies. Protocol Developers need to have a contingency plan in advance for any security problems with the underlying assets. It is essential to prevent a bad asset from causing uncontrollable risk to the whole protocol.

### Status

The risk has been discussed.

# 5. Conclusion

After auditing and analyzing the contract code, SECBIT Labs found some issues to optimize and proposed corresponding suggestions, which have been shown above. The BTC Plus developers have them all fixed in the latest version of the code for the issues found. The risks raised in this report have also been discussed. SECBIT Labs holds that the BTC Plus project has high code quality, detailed documentation, and complete test cases. BTC Plus integrates ERC20 BTC Tokens and provides considerable yields, which helps to provide more DeFi applications and opportunities for BTC and ERC20 BTC Tokens. In particular, the team has innovatively constructed a new type of asset through the interesting positive rebase mechanism. This asset can generate income continuously while anchoring a specific asset. We expect this new type of asset to gain more acceptance and become a common asset soon.

# Disclaimer

SECBIT smart contract audit service assesses the contract's correctness, security, and performability in code quality, logic design, and potential risks. The report is provided "as is", without any warranties about the code practicability, business model, management system's applicability and anything related to the contract adaptation. This audit report is not to be taken as an endorsement of the platform, team, company or investment.

# APPENDIX

## Vulnerability/Risk Level Classification

| Level | Description |
|---|---|
| High | Severely damage the contract's integrity and allow attackers to steal ethers and tokens, or lock ethers inside the contract. |
| Medium | Damage contract's security under given conditions and cause impairment of benefit for stakeholders. |
| Low | Cause no actual impairment to contract. |
| Info | Relevant to practice or rationality could possibly bring risks. |

**SECBIT Labs is devoted to constructing a common-consensus, reliable and ordered blockchain economic entity.**

🌎 http://www.secbit.io

✉ audit@secbit.io

🐦 @secbit_io