CERTIK

Security Assessment

BiFrost BTC Extension

Aug 3rd, 2021

Table of Contents

Summary

This report has been prepared for BiFrost to discover issues and vulnerabilities in the source code of the BiFrost BTC Extension project as well as any contract dependencies that were not part of an officially recognized library. A comprehensive examination has been performed, utilizing Static Analysis and Manual Review techniques.

The auditing process pays special attention to the following considerations:

- Testing the smart contracts against both common and uncommon attack vectors.
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Ensuring contract logic meets the specifications and intentions of the client.
- Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- Thorough line-by-line manual review of the entire codebase by industry experts.

The security assessment resulted in findings that ranged from critical to informational. We recommend addressing these findings to ensure a high level of security standards and industry practices. We suggest recommendations that could better serve the project from the security perspective:

- Enhance general coding practices for better structures of source codes;
- Add enough unit tests to cover the possible use cases;
- Provide more comments per each function for readability, especially contracts that are verified in public;
- Provide more transparency on privileged activities once the protocol is live.

Overview

Project Summary

Project Name	BiFrost BTC Extension
Description	The report comprises the audit of Bifi BTC Extention smart contracts from Bifrost.
Platform	Ethereum
Language	Solidity
Codebase	5084f0ab34493aef48e54400486bf2f1963707f1 fb0567f98bc0d563e2180066fb0b4a4eb845cbcf
Commit	5084f0ab34493aef48e54400486bf2f1963707f1 fb0567f98bc0d563e2180066fb0b4a4eb845cbcf

Audit Summary

Delivery Date	Aug 03, 2021
Audit Methodology	Static Analysis, Manual Review
Key Components	

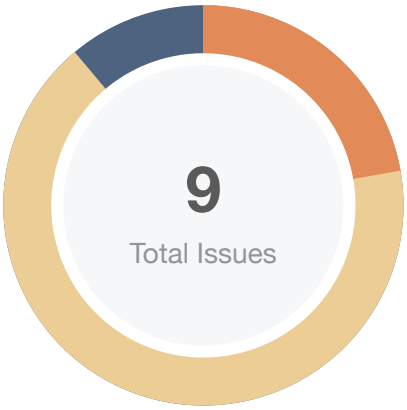
Vulnerability Summary

Vulnerability Level	Total	Pending	Partially Resolved	Resolved	Acknowledged	Declined
Critical	0	0	0	0	0	0
Major	2	0	0	1	1	0
Medium	0	0	0	0	0	0
Minor	6	0	0	3	1	2
Informational	1	0	0	1	0	0
Discussion	0	0	0	0	0	0

Audit Scope

ID	File	SHA256 Checksum
BTA	BTCbase/adapter/BTCTokenAdapter.sol	d5558ccdcd9aa1b69ec0cf362fb2cd38b6bd7cf14b17b2fe278871d28822d346
BCE	BTCbase/events/BTCEvents.sol	c59f38a8c74dd9f94380897eeeaddf797fae889dd2ee8c17123e1c70e8e00326
BTD	BTCbase/state/BTCDataStructure.sol	09d041aa4cebca55120412c5f365d8e9550e25672ac4fa346062d9c9beab10a5
BTS	BTCbase/state/BTCStorage.sol	5d1b9be1791a30d084eacb891444e109e73e9e5c1338a3edea8de96a78f83bd7
BCD	BTCbase/storage/BTCDataStorage.sol	f6ceea53a6d14c6f439cca45953033433372fa4bc856f555d4edf8ecf46b87f8
BMB	BTCbase/utils/BytesModule.sol	857d92ad3371233dba7e9b974ebf908f1de1d1022c34ef84b37360fea54155d0
SDS	BTCbase/utils/SetDataStructure.sol	9502a1114a2c742488eb211925f28d44c5ba768a15803e5b5c95ddc96551a645
BTE	BTCbase/BTCEntryLogicExternal.sol	f3725a0d165ef060d8d37d4d0c2b23670bec3770b895c5b7063bf28591d35c0e
BCI	BTCbase/BTCInternalBase.sol	4a50aa6387a2dab7c646f41d2222730e5df288fbbe0285c851321113ac3ad26a
RSR	Resolverbase/state/ResolverStorage.sol	093f6dba70a4d9b166bfd15ccc845f44345c63f7424d9dd5d44fabb2bd594be7
RSB	Resolverbase/state/ResolverStructure.sol	95d1517e2dc0ad4073dd5a361c6cf1d24091d94bdf0471ad84d2592380121b02
REL	Resolverbase/ResolverEntryLogic.sol	b97911958d37c317dcf0e2735be925af7f4bb9ba76bc3f49f276c1f420a683a6
SPF	ScorePot/state/ScorePotStorage.sol	b9a5d4c05d0e805f8e8d02055b4aa4040e9c0aec52062699d5a1299370453460
SPT	ScorePot/state/ScorePotStructure.sol	ab8b7b8f51a18f6c6c1c48a2b048b09a2054a8cf9aeb6e49d163da48ef7580aa
SPS	ScorePot/ScorePot.sol	8972172f3a160a13f7dbf796f062c631a50685bc788b6fbe0e390b0945680a35
SPP	ScorePot/ScorePotInternal.sol	64b6288cb13734d6ee79f3b3a6ca060bfb277361c4bffd88c4d831fc985c46bc

Findings



Critical	0 (0.00%)
Major	2 (22.22%)
Medium	0 (0.00%)
Minor	6 (66.67%)
Informational	1 (11.11%)
Discussion	0 (0.00%)

ID	Title	Category	Severity	Status
BCD-01	Inexistent Visibility Specifiers	Coding Style	Informational	Resolved
BMB-01	Redundant Slice Implementation	Gas Optimization	Minor	Declined
BTA-01	Unchecked ERC-20 Transfers	Logical Issue	Minor	Acknowledged
BTS-01	Inexistent Access Control	Logical Issue	Major	Resolved
BTS-02	Unsafe Approval Amount	Logical Issue	Minor	Declined
SDS-01	Convoluted Implementation	Logical Issue	Minor	Resolved
SPP-01	Flash Loan Susceptibility	Logical Issue	Major	Acknowledged
SPS-01	Unchecked ERC-20 Transfers	Logical Issue	Minor	Resolved
SPS-02	Inexplicable Functionality	Logical Issue	Minor	Resolved

BCD-01 | Inexistent Visibility Specifiers

Category	Severity	Location	Status
Coding Style	Informational	BTCbase/storage/BTCDataStorage.sol: 16, 19, 20, 25, 26, 30, 34~36	Resolved

Description

The linked variables do not have a visibility specifier set.

Recommendation

We advise one to be set so.

Alleviation

Alleviations are applied as of commit hash `fb0567f98bc0d563e2180066fb0b4a4eb845cbcf`.

BMB-01 | Redundant Slice Implementation

Category	Severity	Location	Status
Gas Optimization	Minor	BTCbase/utils/BytesModule.sol: 31~66	Declined

Description

Ever since Solidity `0.6.0`, Solidity has built-in support for slices when applied on `calldata` arguments.

Recommendation

We advise the function to be omitted and the built-in usage to be performed. The only segment in the codebase using the `slice` function is within `_checkHeaderChain` of `BTCEntryLogicInteral` which passes in the `rawHeader` calldata argument from `BTCEntryLogicExternal`. As such, this type of adjustment can be performed and ensures a higher degree of safety as well as optimization in the codebase.

Alleviation

The team did not consider the recommendation and stated that "Unfortunately, the handled (sliced) in the internal logic code, because it refers other values that are accessible in the internal logic. Moreover, the function cannot be used to the calldata in the internal memory variable. Thus, we have used the old-fashioned way to slice the calldata.".

BTA-01 | Unchecked ERC-20 Transfers

Category	Severity	Location	Status
Logical Issue	Minor	BTCbase/adapter/BTCTokenAdapter.sol: 36, 42	Acknowledged

Description

The `BTCTokenAdapter` contract does not properly evaluate the returned `bool` of ERC-20 transfers.

Recommendation

We advise the value to be handled opportunistically by introducing the `SafeERC20` library from OpenZeppelin and utilizing that.

Alleviation

The team did not consider the recommendation and stated that "The ERC-20 transfer is performed only with BFC and BTC-wrapping token token contract. Both two token contracts are standard ERC-20 token contracts. If there is an error cases then there will be operation. Thus, we don't need additional condition to check the return value of the transfer. Please note that, due to the bytecode size limit, it is hard to add more library, such as SafeERC20.".

BTS-01 | Inexistent Access Control

Category	Severity	Location	Status
Logical Issue	Major	BTCbase/state/BTCStorage.sol: 60~64	Resolved

Description

The `initBitcoinState` has no access control imposed on it permitting anyone to invoke it and reset states.

Recommendation

We advise some form of access control to be imposed on it.

Alleviation

The code part under consideration is commented as of commit hash `fb0567f98bc0d563e2180066fb0b4a4eb845cbcf` rendering the finding ineffectual.

BTS-02 | Unsafe Approval Amount

Category	Severity	Location	Status
Logical Issue	Minor	BTCbase/state/BTCStorage.sol: 74	Declined

Description

The `setHandler` function perfroms an unlimited approval of `BWBTC` to the `handlerAddr` which should be deemed unsafe.

Recommendation

We advise another approval system to be utilized instead whereby an exact value is approved or approval is requested by the handler itself.

Alleviation

The responded that "The Bitcoin contract connects with our handler contact (i.e., Bitcoin handler) only. We are aware the risk of unlimited approval, but these are contacts consisting the BiFi service.".

SDS-01 | Convoluted Implementation

Category	Severity	Location	Status
Logical Issue	Minor	BTCbase/utils/SetDataStructure.sol: 16~139	Resolved

Description

The implementation of the set appears incorrectly done so given that certain insertion mechanisms perform inexplicable statements, such as the `push` instruction in the case of `ptr.count < ptr.size`.

Recommendation

We advise an implementation such as the one offered by OpenZeppelin to be utilized given that the order via which the elements are sorted in this set is convoluted. Alternatively, we advise the system to be revamped to instead utilize actual index pointers to the elements rather than a value based approach as a data that is equal to an existing one during insertion will cause misbehaviours.

Alleviation

Alleviations are applied as of commit hash `fb0567f98bc0d563e2180066fb0b4a4eb845cbcf` and the team stated that "We had named the data structure incorrectly. The data structure is not for "set", but for implementing checkpoints of Bitcoin transaction relays.".

SPP-01 | Flash Loan Susceptibility

Category	Severity	Location	Status
Logical Issue	Major	ScorePot/ScorePotInternal.sol: 14	Acknowledged

Description

The `ScorePotInternal` relies on the spot price of BFC/ETH which opens up a flash-loan manipulation attack vector.

Recommendation

We advise some other form of price oracle to be utilized, potentially akin to a TWAP.

Alleviation

The team did not consider the recommendation at the moment and stated that "It's a design decision. BFC is used only for paying fees. Thus, the impact of possible Flash Loan attack is limited to the fee calculation. We will update the code to use stable price feed (e.g., Chainlink) once the BFC price is listed on the Chainlink price feed.".

SPS-01 | Unchecked ERC-20 Transfers

Category	Severity	Location	Status
Logical Issue	Minor	ScorePot/ScorePot.sol: 16, 21, 25, 26	Resolved

Description

The `ScorePot` contract does not properly evaluate the returned `bool` of ERC-20 transfers.

Recommendation

We advise the value to be handled opportunistically by introducing the `SafeERC20` library from OpenZeppelin and utilizing that.

Alleviation

The code part under consideration is removed as of commit hash `fb0567f98bc0d563e2180066fb0b4a4eb845cbcf` rendering this finding ineffectual.

SPS-02 | Inexplicable Functionality

Category	Severity	Location	Status
Logical Issue	Minor	ScorePot/ScorePot.sol: 95	Resolved

Description

The administrator of the contract is able to delete the total score without necessarily updating the existing scorers of the contract.

Recommendation

As deleting the total score independently is not a logical state transition, we advise these statements to be bundled in a utility function similarly to `pop_scorers`.

Alleviation

The code part under consideration is removed as of commit hash `fb0567f98bc0d563e2180066fb0b4a4eb845cbcf` rendering this finding ineffectual and the team stated that "The corresponding code parts are not used anymore and will be removed when we launch the service.".

Appendix

Finding Categories

Gas Optimization

Gas Optimization findings do not affect the functionality of the code but generate different, more optimal EVM opcodes resulting in a reduction on the total gas cost of a transaction.

Logical Issue

Logical Issue findings detail a fault in the logic of the linked code, such as an incorrect notion on how block.timestamp works.

Coding Style

Coding Style findings usually do not affect the generated byte-code but rather comment on how to make the codebase more legible and, as a result, easily maintainable.

Checksum Calculation Method

The "Checksum" field in the "Audit Scope" section is calculated as the SHA-256 (Secure Hash Algorithm 2 with digest size of 256 bits) digest of the content of each file hosted in the listed source repository under the specified commit.

The result is hexadecimal encoded and is the same as the output of the Linux "sha256sum" command against the target file.

Disclaimer

This report is subject to the terms and conditions (including without limitation, description of services, confidentiality, disclaimer and limitation of liability) set forth in the Services Agreement, or the scope of services, and terms and conditions provided to the Company in connection with the Agreement. This report provided in connection with the Services set forth in the Agreement shall be used by the Company only to the extent permitted under the terms and conditions set forth in the Agreement. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes without CertiK's prior written consent.

This report is not, nor should be considered, an "endorsement" or "disapproval" of any particular project or team. This report is not, nor should be considered, an indication of the economics or value of any "product" or "asset" created by any team or project that contracts CertiK to perform a security assessment. This report does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors, business, business model or legal compliance.

This report should not be used in any way to make decisions around investment or involvement with any particular project. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Blockchain technology and cryptographic assets present a high level of ongoing risk. CertiK's position is that each company and individual are responsible for their own due diligence and continuous security. CertiK's goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies, and in no way claims any guarantee of security or functionality of the technology we agree to analyze.

About

Founded in 2017 by leading academics in the field of Computer Science from both Yale and Columbia University, CertiK is a leading blockchain security company that serves to verify the security and correctness of smart contracts and blockchain-based protocols. Through the utilization of our world-class technical expertise, alongside our proprietary, innovative tech, we're able to support the success of our clients with best-in-class security, all whilst realizing our overarching vision; provable trust for all throughout all facets of blockchain.