IRON BANK SECURITY AUDIT REPORT

September 5, 2022

MixBytes()

TABLE OF CONTENTS

1. INTRODUCTION

1.1 Disclaimer

The audit makes no statements or warranties about utility of the code, safety of the code, suitability of the business model, investment advice, endorsement of the platform or its products, regulatory regime for the business model, or any other statements about fitness of the contracts to purpose, or their bug free status. The audit documentation is for discussion purposes only.

1.2 Security Assessment Methodology

A group of auditors are involved in the work on the audit. The security engineers check the provided source code independently of each other in accordance with the methodology described below:

1. Project architecture review:

- Project documentation review.
- General code review.
- Reverse research and study of the project architecture on the source code alone.

Stage goals

- Build an independent view of the project's architecture.
- Identifying logical flaws.

2. Checking the code in accordance with the vulnerabilities checklist:

- Manual code check for vulnerabilities listed on the Contractor's internal checklist. The Contractor's checklist is constantly updated based on the analysis of hacks, research, and audit of the clients' codes.
- Code check with the use of static analyzers (i.e Slither, Mythril, etc).

Stage goal

Eliminate typical vulnerabilities (e.g. reentrancy, gas limit, flash loan attacks etc.).

3. Checking the code for compliance with the desired security model:

- Detailed study of the project documentation.
- Examination of contracts tests.
- Examination of comments in code.
- Comparison of the desired model obtained during the study with the reversed view obtained during the blind audit.
- Exploits PoC development with the use of such programs as Brownie and Hardhat.

Stage goal

Detect inconsistencies with the desired model.

4. Consolidation of the auditors' interim reoprts into one:

- Cross check: each auditor reviews the reports of the others.
- Discussion of the issues found by the auditors.
- Issuance of an interim audit report.

Stage goals

- Double-check all the found issues to make sure they are relevant and the determined threat level is correct.
- Provide the Customer with an interim report.

5. Bug fixing & re-audit:

- The Customer either fixes the issues or provides comments on the issues found by the auditors. Feedback from the Customer must be received on every issue/bug so that the Contractor can assign them a status (either "fixed" or "acknowledged").
- Upon completion of the bug fixing, the auditors double-check each fix and assign it a specific status, providing a proof link to the fix.
- A re-audited report is issued.

Stage goals

- Verify the fixed code version with all the recommendations and its statuses.
- Provide the Customer with a re-audited report.

6. Final code verification and issuance of a public audit report:

- The Customer deploys the re-audited source code on the mainnet.
- The Contractor verifies the deployed code with the re-audited version and checks them for compliance.
- If the versions of the code match, the Contractor issues a public audit report.

Stage goals

- Verify the fixed code version with all the recommendations and its statuses.
- Provide the Customer with a re-audited report.

Finding Severity breakdown

All vulnerabilities discovered during the audit are classified based on their potential severity and have the following classification:

Severity	Description
Critical	Bugs leading to assets theft, fund access locking, or any other loss funds to be transferred to any party.
High	Bugs that can trigger a contract failure. Further recovery is possible only by manual modification of the contract state or replacement.
Medium	Bugs that can break the intended contract logic or expose it to DoS attacks, but do not cause direct loss funds.
Low	Bugs that do not have a significant immediate impact and could be easily fixed.

Based on the feedback received from the Customer regarding the list of findings discovered by the Contractor, they are assigned the following statuses:

Status	Description
Fixed	Recommended fixes have been made to the project code and no longer affect its security.
Acknowledged	The Customer is aware of the finding. Recommendations for the finding are planned to be resolved in the future.

1.3 Project Overview

IronBank is a fork of Compound Protocol with some functionality improvement, such as ability to provide uncollaterized credit to external protocols, supporting of Chainlink and Band price feed and some architecture improvements.

1.4 Project Dashboard

Project Summary

Title	Description
Client	IronBank
Project name	IronBank
Timeline	01-08-2022 - 05-09-2022
Number of Auditors	3

Project Log

Date	Commit Hash	Note
05.08.2019	f385d71983ae5c5799faae9b2dfea43e5cf75262	Audited by Trail of Bits, compound-3.pdf
01.08.2022	8cd45803b48552e344e22be280c9e1c03ec8644a	Changes from previous audit
22.08.2022	5581327fc855c2734d65a5dd8f198bca6f8963e8	Commit with fixes of findings discovered by MixBytes
29.08.2022	4883f8a6d6faeafa82f7b1979cd77c8cc2b59b5c	Final commit

Project Scope

The audit covered the following files:

File name	Link
CCollateralCapErc20.sol	CCollateralCapErc20.sol
CCollateralCapErc20CheckRepay.sol	CCollateralCapErc20CheckRepay.sol
CCollateralCapErc20CheckRepayDelegate.sol	CCollateralCapErc20CheckRepayDelegate.sol
CCollateralCapErc20NoInterest.sol	CCollateralCapErc20NoInterest.sol
CCollateralCapErc20NoInterestDelegate.sol	CCollateralCapErc20NoInterestDelegate.sol
CErc20.sol	CErc20.sol
CErc20Delegate.sol	CErc20Delegate.sol
CErc20Delegator.sol	CErc20Delegator.sol
CToken.sol	CToken.sol
CTokenAdmin.sol	CTokenAdmin.sol
CTokenCheckRepay.sol	CTokenCheckRepay.sol
CTokenInterfaces.sol	CTokenInterfaces.sol
CTokenNoInterest.sol	CTokenNoInterest.sol
CWrappedNative.sol	CWrappedNative.sol
CWrappedNativeDelegate.sol	CWrappedNativeDelegate.sol
CWrappedNativeDelegator.sol	CWrappedNativeDelegator.sol
CarefulMath.sol	CarefulMath.sol
Comptroller.sol	Comptroller.sol

File name	Link
ComptrollerInterface.sol	ComptrollerInterface.sol
ComptrollerStorage.sol	ComptrollerStorage.sol
EIP20Interface.sol	EIP20Interface.sol
EIP20NonStandardInterface.sol	EIP20NonStandardInterface.sol
ERC3156FlashBorrowerInterface.sol	ERC3156FlashBorrowerInterface.sol
ERC3156FlashLenderInterface.sol	ERC3156FlashLenderInterface.sol
ErrorReporter.sol	ErrorReporter.sol
Exponential.sol	Exponential.sol
InterestRateModel.sol	InterestRateModel.sol
JumpRateModelV2.sol	JumpRateModelV2.sol
LiquidityMiningInterface.sol	LiquidityMiningInterface.sol
Maximillion.sol	Maximillion.sol
SafeMath.sol	SafeMath.sol
TripleSlopeRateModel.sol	TripleSlopeRateModel.sol
Unitroller.sol	Unitroller.sol
Denominations.sol	Denominations.sol
PriceOracle.sol	PriceOracle.sol
PriceOracleProxy.sol	PriceOracleProxy.sol
PriceOracleProxyIB.sol	PriceOracleProxyIB.sol
SimplePriceOracle.sol	SimplePriceOracle.sol

File name	Link
v1PriceOracle.sol	v1PriceOracle.sol
CompoundLens.sol	CompoundLens.sol
Comp.sol	Comp.sol

1.5 Summary of findings

Severity	# of Findings
Critical	0
High	0
Medium	5
Low	2

ID	Name	Severity	Status
M-1	Set credit limit by pausing the guardian	Medium	Fixed
M-2	Exchange rate vulnerability	Medium	Acknowledged
M-3	Interest rate model update impacts the old time period	Medium	Fixed
M-4	A flashloan will be broken if the USDT fee is more than zero	Medium	Acknowledged
M-5	Undesired repay and/or liquidation of ex-credit account	Medium	Fixed

L-1	Typos in descriptions	Low	Fixed
L-2	No null checks for input addresses	Low	Fixed

1.6 Conclusion

During the audit, 5 findings of medium severity were identified and confirmed by the Client. The Client have fixed 3 issues of medium severity, and 2 issues have been acknowledged. Those issues do not have a significant impact and can be resolved through careful deployment and maintenance procedures.

File name	Contract deployed on mainnet
CToken.sol	0xcB9Ab119BE270F58d40e3D57D1ecC82bd479D59F
CarefulMath.sol	0xcB9Ab119BE270F58d40e3D57D1ecC82bd479D59F
Comptroller.sol	0xcB9Ab119BE270F58d40e3D57D1ecC82bd479D59F
ComptrollerStorage.sol	0xcB9Ab119BE270F58d40e3D57D1ecC82bd479D59F
ErrorReporter.sol	0xcB9Ab119BE270F58d40e3D57D1ecC82bd479D59F
Exponential.sol	0xcB9Ab119BE270F58d40e3D57D1ecC82bd479D59F
Comp.sol	0xcB9Ab119BE270F58d40e3D57D1ecC82bd479D59F
InterestRateModel.sol	0xcB9Ab119BE270F58d40e3D57D1ecC82bd479D59F
LiquidityMiningInterface.sol	0xcB9Ab119BE270F58d40e3D57D1ecC82bd479D59F
PriceOracle.sol	0xcB9Ab119BE270F58d40e3D57D1ecC82bd479D59F
Unitroller.sol	0xcB9Ab119BE270F58d40e3D57D1ecC82bd479D59F
CErc20.sol	0xD5734c42E2e593933231bE61BAc2B94ACdc44DC4
CToken.sol	0xD5734c42E2e593933231bE61BAc2B94ACdc44DC4

File name	Contract deployed on mainnet
CarefulMath.sol	0xD5734c42E2e593933231bE61BAc2B94ACdc44DC4
ComptrollerStorage.sol	0xD5734c42E2e593933231bE61BAc2B94ACdc44DC4
ErrorReporter.sol	0xD5734c42E2e593933231bE61BAc2B94ACdc44DC4
Exponential.sol	0xD5734c42E2e593933231bE61BAc2B94ACdc44DC4
InterestRateModel.sol	0xD5734c42E2e593933231bE61BAc2B94ACdc44DC4
Denominations.sol	0xD5734c42E2e593933231bE61BAc2B94ACdc44DC4
PriceOracle.sol	0xD5734c42E2e593933231bE61BAc2B94ACdc44DC4
PriceOracleProxyIB.sol	0xD5734c42E2e593933231bE61BAc2B94ACdc44DC4
TripleSlopeRateModel.sol (Gov)	0xd0B628cB062bcb34331482391C2110CD7a731e5a
SafeMath.sol	0xd0B628cB062bcb34331482391C2110CD7a731e5a
TripleSlopeRateModel.sol (Stable)	0x8015272057745533Fc531B6429c2d2F51BE3711C
TripleSlopeRateModel.sol (Major)	0xadc46C5eA23BcB838Af714bCD822d5f52c2EDF23
TripleSlopeRateModel.sol (WETH)	0xf2Acee535Ebb8B9Bb875646E134CdDEb6f5a97ef

2.FINDINGS REPORT

2.1 Critical

Not Found

2.2 High

Not Found

2.3 Medium

M-1	Set credit limit by pausing the guardian
File	Comptroller.sol#L1313
Severity	Medium
Status	Fixed in 66ca6047

Description

Pausing the guardian can set a new credit limit for users with the credit limit. Comptroller.sol#L1313

Recommendation

We recommend updating the checks in the "_setCreditLimit" function.

M-2	Exchange rate vulnerability
File	
Severity	Medium
Status	Acknowledged

Description

An exchange rate bug for new pools and empty pools (without borrowers and suppliers) for CToken contracts without the 'internalCash' variable.
Flow:

1. Create cToken
2. Mint cToken by user1 (1,000,000)
3. Redeem cToken by user1 (999,999.999999)
4. Transfer underlying (1,000,000) from user1 to market
5. Mint cToken by user2 (1,000,000)
6. Redeem cToken by user1 (user1 receive extra tokens)

Recommendation

We recommend checking the exchange rate before the first mint or using the 'internalCash' value for all CToken contracts.

Client's commentary

No need to fix. All new pools will be deployed with CCollateralCap implementation which has internalCash to prevent such vulnerability.

M-3	Interest rate model update impacts the old time period
File	TripleSlopeRateModel.sol#L100
Severity	Medium
Status	Fixed in 2773bdaa

Description

After an admin changes the interest rate model parameters by using this function TripleSlopeRateModel.sol#L100
indexes will be recalculated in the upcoming accrueInterest() function call. But this call applies new interest settings to the previous period of time which is not correct.

Recommendation

The interest rate model parameters should be changed just after calling the accrueInterest() function for each asset. It can be done by creating a special service contract.

M-4	A flashloan will be broken if the USDT fee is more than zero
File	CCollateralCapErc20.sol#L217
Severity	Medium
Status	Acknowledged

Description

Let's take a look at the flashloan flow. After doTransferOut a receiver gets `amount - fee`.
CCollateralCapErc20.sol#L217

Then a receiver's `onFlashLoan` function will be called with an incorrect amount.
CCollateralCapErc20.sol#L224

Then doTransferIn will transfer the repayment amount but the contract will receive the repayment `amount - fee`
CCollateralCapErc20.sol#L231 and the`require` check will cause a revert.
CCollateralCapErc20.sol#L235

Recommendation

The flashloan() function should be rewritten taking into consideration the USDT fee value.

Client's commentary

No need to fix. Currently there is no USDT fee value. Since it affects Flash Loan only, we consider it acceptable to fix the issue reactively along with Compound.

M-5	Undesired repay and/or liquidation of ex-credit account
File	Comptroller.sol#L1302
Severity	Medium
Status	Fixed in 66ca6047

Description

In the IronBank, the `credit` functionality is introduced. An admin can trigger the Comptroller.sol#L1302 function to mark some addresses like `credit` for specific `cToken` and set its `credit limit`. Such addresses can borrow a limited by `credit limit` amount of `cToken` without providing any collateral. Additionally, `setCreditLimit` can mark that an address is no longer `credit` and has become an ordinary account that requires collateral.

Unfortunately, after becoming an ordinary account, the ex-credit account will be subject to repay and/or liquidation of its borrowed debt.

Recommendation

Although an attack is hard to implement since the `setCreditLimit` function is restricted to the admin, we recommend to disallow changing the state from the credit account to an ordinary account, e.g. by disallowing setting the credit limit less than the currently borrowed amount.

2.4 Low

L-1	Typos in descriptions
Files	CTokenInterfaces.sol#L366 PriceOracleProxy.sol#L160 PriceOracleProxyIB.sol#L96 Comptroller.sol#L1144 CErc20Delegator.sol#L54 CCollateralCapErc20.sol#L556 CCollateralCapErc20CheckRepay.sol#L557 CCollateralCapErc20NoInterest.sol#L557 CErc20.sol#L300 CWrappedNative.sol#L659 ComptrollerStorage.sol#L100 CToken.sol#L540 CTokenCheckRepay.sol#L540 CTokenNoInterest.sol#L569 InterestRateModel.sol#L15
Severity	Low
Status	Fixed in a059518d

Description

Several typos:

1. 'occured' instead of 'occurred'
   CTokenInterfaces.sol#L366
   CTokenInterfaces.sol#L412
2. 'fucntions' instead of 'functions'
   PriceOracleProxy.sol#L160
   PriceOracleProxyIB.sol#L96
3. 'supplys' instead of 'supplies'
   Comptroller.sol#L1144
4. 'currenlty' instead of 'currently'
   PriceOracleProxy.sol#L189
   PriceOracleProxyIB.sol#L101
5. 'settor' instead of 'setter'
   CErc20Delegator.sol#L54
6. 'accuring' instead of 'accruing'
   CCollateralCapErc20.sol#L556
   CCollateralCapErc20.sol#L595
   CCollateralCapErc20CheckRepay.sol#L557

CCollateralCapErc20CheckRepay.sol#L596
CCollateralCapErc20NoInterest.sol#L557
CCollateralCapErc20NoInterest.sol#L596
CErc20.sol#L300
CErc20.sol#L405
CErc20.sol#L475
CWrappedNative.sol#L659
CWrappedNative.sol#L729

7. 'sieze' instead of 'seize'
   Comptroller.sol#L595
8. 'depreacted' instead of 'deprecated'
   ComptrollerStorage.sol#L100
   ComptrollerStorage.sol#L104
   ComptrollerStorage.sol#L108
   ComptrollerStorage.sol#L112
   ComptrollerStorage.sol#L116
   ComptrollerStorage.sol#L120
   ComptrollerStorage.sol#L136
9. 'undelrying' instead of 'underlying'
   CToken.sol#L540
   CTokenCheckRepay.sol#L540
   CTokenNoInterest.sol#L569
10. 'tather' instead of 'rather'
    CToken.sol#L1040
    CTokenCheckRepay.sol#L1059
11. 'amnount' instead of 'amount'
    InterestRateModel.sol#L15
    InterestRateModel.sol#L28

Recommendation

We recommend correcting them.

L-2	No null checks for input addresses
File	PriceOracleProxyIB.sol#L157
Severity	Low
Status	Fixed in f90921f0

Description

Some code lacks a check for null address:
admin: PriceOracleProxyIB.sol#L157

Recommendation

We recommend adding null checks.

3. ABOUT MIXBYTES

MixBytes is a team of blockchain developers, auditors and analysts keen on decentralized systems. We build opensource solutions, smart contracts and blockchain protocols, perform security audits, work on benchmarking and software testing solutions, do research and tech consultancy.

Contacts

https://github.com/mixbytes/audits_public

https://mixbytes.io/

hello@mixbytes.io

https://twitter.com/mixbytes