GEARBOX PROTOCOL SMART CONTRACT AUDIT

December 23, 2021

MixBytes()

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. The information presented in this report is confidential and privileged. If you are reading this report, you agree to keep it confidential, not to copy, disclose or disseminate without the agreement of GearBox Protocol . If you are not the intended recipient(s) of this document, please note that any disclosure, copying or dissemination of its content is strictly forbidden.

1.2 SECURITY ASSESSMENT METHODOLOGY

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

01 Project architecture review:
> Reviewing project documentation
> General code review
> Reverse research and study of the architecture of the code based on the source code only
> Mockup prototyping

Stage goal:
Building an independent view of the project's architecture and identifying logical flaws in the code.

02 Checking the code against the checklist of known vulnerabilities:
> Manual code check for vulnerabilities from the company's internal checklist
> The company's checklist is constantly updated based on the analysis of hacks, research and audit of the clients' code
> Checking with static analyzers (i.e Slither, Mythril, etc.)

Stage goal:
Eliminate typical vulnerabilities (e.g. reentrancy, gas limit, flashloan attacks, etc.)

03 Checking the code for compliance with the desired security model:
> Detailed study of the project documentation
> Examining contracts tests
> Examining comments in code
> Comparison of the desired model obtained during the study with the reversed view obtained during the blind audit
> Exploits PoC development using Brownie

Stage goal:
Detection of inconsistencies with the desired model

04 Consolidation of interim auditor reports into a general one:
> Cross-check: each auditor reviews the reports of the others
> Discussion of the found issues by the auditors
> Formation of a general (merged) report

Stage goal:
Re-check all the problems for relevance and correctness of the threat level and provide the client with an interim report.

05 Bug fixing & re-check:
> Client fixes or comments on every issue
> Upon completion of the bug fixing, the auditors double-check each fix and set the statuses with a link to the fix

Stage goal:
Preparation of the final code version with all the fixes

06 Preparation of the final audit report and delivery to the customer.

Findings discovered during the audit are classified as follows:

FINDINGS SEVERITY BREAKDOWN

Level	Description	Required action
Critical	Bugs leading to assets theft, fund access locking, or any other loss funds to be transferred to any party	Immediate action to fix issue
Major	Bugs that can trigger a contract failure. Further recovery is possible only by manual modification of the contract state or replacement.	Implement fix as soon as possible
Warning	Bugs that can break the intended contract logic or expose it to DoS attacks	Take into consideration and implement fix in certain period
Comment	Other issues and recommendations reported to/acknowledged by the team	Take into consideration

Based on the feedback received from the Customer's team 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 project team is aware of this finding. Recommendations for this finding are planned to be resolved in the future. This finding does not affect the overall safety of the project.
No issue	Finding does not affect the overall safety of the project and does not violate the logic of its work.

1.3 PROJECT OVERVIEW

Gearbox is a generalized leverage protocol which allows both individual users and platforms to increase their capital efficiency. Open a Credit Account and connect to various DeFi protocols in a composable manner: margin trade on Uniswap, leverage farm on Yearn and Curve, and more.Gearbox enables undercollateralized interactions with external DeFi protocols through margin lending in a composable manner. Instead of going for credit scoring, Gearbox introduces Credit Accounts - a DeFi primitive which allows users to execute financial orders without accessing funds on it, such as that account acts as collateral for different undercollateralized operations.

1.4 PROJECT DASHBOARD

Client	GearBox Protocol
Audit name	GearBox Protocol
Initial version	0ac33ba87212ce056ac6b6357ad74161d417158a
Final version	7ceb7807af8585bff65387054fe5ded5e66bbfcf
Date	August 31, 2021 - December 23, 2021
Auditors engaged	4 auditors

FILES LISTING

CurveV1.sol	https://github.com/Gearbox-protocol/gearbox-contracts/blob/0ac33ba87212ce056ac6b6357ad74161d417158a/contracts/adapters/CurveV1.sol
UniswapV2.sol	https://github.com/Gearbox-protocol/gearbox-contracts/blob/0ac33ba87212ce056ac6b6357ad74161d417158a/contracts/adapters/UniswapV2.sol
UniswapV3.sol	https://github.com/Gearbox-protocol/gearbox-contracts/blob/0ac33ba87212ce056ac6b6357ad74161d417158a/contracts/adapters/UniswapV3.sol
YearnV2.sol	https://github.com/Gearbox-protocol/gearbox-contracts/blob/0ac33ba87212ce056ac6b6357ad74161d417158a/contracts/adapters/YearnV2.sol
ACLTrait.sol	https://github.com/Gearbox-protocol/gearbox-contracts/blob/0ac33ba87212ce056ac6b6357ad74161d417158a/contracts/core/ACLTrait.sol

ACL.sol	https://github.com/Gearbox-protocol/gearbox-contracts/blob/0ac33ba87212ce056ac6b6357ad74161d417158a/contracts/core/ACL.sol
AccountFactory.sol	https://github.com/Gearbox-protocol/gearbox-contracts/blob/0ac33ba87212ce056ac6b6357ad74161d417158a/contracts/core/AccountFactory.sol
AccountMining.sol	https://github.com/Gearbox-protocol/gearbox-contracts/blob/0ac33ba87212ce056ac6b6357ad74161d417158a/contracts/core/AccountMining.sol
AddressProvider.sol	https://github.com/Gearbox-protocol/gearbox-contracts/blob/0ac33ba87212ce056ac6b6357ad74161d417158a/contracts/core/AddressProvider.sol
ContractsRegister.sol	https://github.com/Gearbox-protocol/gearbox-contracts/blob/0ac33ba87212ce056ac6b6357ad74161d417158a/contracts/core/ContractsRegister.sol
DataCompressor.sol	https://github.com/Gearbox-protocol/gearbox-contracts/blob/0ac33ba87212ce056ac6b6357ad74161d417158a/contracts/core/DataCompressor.sol
WETHGateway.sol	https://github.com/Gearbox-protocol/gearbox-contracts/blob/0ac33ba87212ce056ac6b6357ad74161d417158a/contracts/core/WETHGateway.sol
CreditAccount.sol	https://github.com/Gearbox-protocol/gearbox-contracts/blob/0ac33ba87212ce056ac6b6357ad74161d417158a/contracts/credit/CreditAccount.sol
CreditFilter.sol	https://github.com/Gearbox-protocol/gearbox-contracts/blob/0ac33ba87212ce056ac6b6357ad74161d417158a/contracts/credit/CreditFilter.sol
CreditManager.sol	https://github.com/Gearbox-protocol/gearbox-contracts/blob/0ac33ba87212ce056ac6b6357ad74161d417158a/contracts/credit/CreditManager.sol
LeverageActions.sol	https://github.com/Gearbox-protocol/gearbox-contracts/blob/0ac33ba87212ce056ac6b6357ad74161d417158a/contracts/credit/LeverageActions.sol
ICurvePool.sol	https://github.com/Gearbox-protocol/gearbox-contracts/blob/0ac33ba87212ce056ac6b6357ad74161d417158a/contracts/integrations/curve/ICurvePool.sol
BytesLib.sol	https://github.com/Gearbox-protocol/gearbox-contracts/blob/0ac33ba87212ce056ac6b6357ad74161d417158a/contracts/integrations/uniswap/BytesLib.sol
IQuoter.sol	https://github.com/Gearbox-protocol/gearbox-contracts/blob/0ac33ba87212ce056ac6b6357ad74161d417158a/contracts/integrations/uniswap/IQuoter.sol

IUniswapV2Migrator.sol	https://github.com/Gearbox-protocol/gearbox-contracts/blob/0ac33ba87212ce056ac6b6357ad74161d417158a/contracts/integrations/uniswap/IUniswapV2Migrator.sol
IUniswapV2Router01.sol	https://github.com/Gearbox-protocol/gearbox-contracts/blob/0ac33ba87212ce056ac6b6357ad74161d417158a/contracts/integrations/uniswap/IUniswapV2Router01.sol
IUniswapV2Router02.sol	https://github.com/Gearbox-protocol/gearbox-contracts/blob/0ac33ba87212ce056ac6b6357ad74161d417158a/contracts/integrations/uniswap/IUniswapV2Router02.sol
IUniswapV3SwapCallback.sol	https://github.com/Gearbox-protocol/gearbox-contracts/blob/0ac33ba87212ce056ac6b6357ad74161d417158a/contracts/integrations/uniswap/IUniswapV3SwapCallback.sol
IUniswapV3.sol	https://github.com/Gearbox-protocol/gearbox-contracts/blob/0ac33ba87212ce056ac6b6357ad74161d417158a/contracts/integrations/uniswap/IUniswapV3.sol
IYVault.sol	https://github.com/Gearbox-protocol/gearbox-contracts/blob/0ac33ba87212ce056ac6b6357ad74161d417158a/contracts/integrations/yearn/IYVault.sol
YearnPriceFeed.sol	https://github.com/Gearbox-protocol/gearbox-contracts/blob/0ac33ba87212ce056ac6b6357ad74161d417158a/contracts/integrations/yearn/YearnPriceFeed.sol
Types.sol	https://github.com/Gearbox-protocol/gearbox-contracts/blob/0ac33ba87212ce056ac6b6357ad74161d417158a/contracts/libraries/data/Types.sol
Constants.sol	https://github.com/Gearbox-protocol/gearbox-contracts/blob/0ac33ba87212ce056ac6b6357ad74161d417158a/contracts/libraries/helpers/Constants.sol
Errors.sol	https://github.com/Gearbox-protocol/gearbox-contracts/blob/0ac33ba87212ce056ac6b6357ad74161d417158a/contracts/libraries/helpers/Errors.sol
PercentageMath.sol	https://github.com/Gearbox-protocol/gearbox-contracts/blob/0ac33ba87212ce056ac6b6357ad74161d417158a/contracts/libraries/math/PercentageMath.sol
WadRayMath.sol	https://github.com/Gearbox-protocol/gearbox-contracts/blob/0ac33ba87212ce056ac6b6357ad74161d417158a/contracts/libraries/math/WadRayMath.sol
PriceOracle.sol	https://github.com/Gearbox-protocol/gearbox-contracts/blob/0ac33ba87212ce056ac6b6357ad74161d417158a/contracts/oracles/PriceOracle.sol

LinearInterestRateModel.sol	https://github.com/Gearbox-protocol/gearbox-contracts/blob/0ac33ba87212ce056ac6b6357ad74161d417158a/contracts/pool/LinearInterestRateModel.sol
PoolService.sol	https://github.com/Gearbox-protocol/gearbox-contracts/blob/0ac33ba87212ce056ac6b6357ad74161d417158a/contracts/pool/PoolService.sol
DieselToken.sol	https://github.com/Gearbox-protocol/gearbox-contracts/blob/0ac33ba87212ce056ac6b6357ad74161d417158a/contracts/tokens/DieselToken.sol
GearNFT.sol	https://github.com/Gearbox-protocol/gearbox-contracts/blob/0ac33ba87212ce056ac6b6357ad74161d417158a/contracts/tokens/GearNFT.sol
GearToken.sol	https://github.com/Gearbox-protocol/gearbox-contracts/blob/0ac33ba87212ce056ac6b6357ad74161d417158a/contracts/tokens/GearToken.sol
Vesting.sol	https://github.com/Gearbox-protocol/gearbox-contracts/blob/0ac33ba87212ce056ac6b6357ad74161d417158a/contracts/tokens/Vesting.sol

FINDINGS SUMMARY

Level	Amount
Critical	1
Major	17
Warning	47
Comment	23

CONCLUSION

Smart contracts have been audited and some suspicious places have been spotted. During the audit 1 critical issue was found and reported to the client. 17 issues were marked as major because they could lead to some undesired behavior, also some warnings and comments were found and discussed with the client. After working on the reported findings most of them were resolved or acknowledged (if the problem was not critical) by the client.Final commit identifier with all fixes:
`7ceb7807af8585bff65387054fe5ded5e66bbfcf`

CONTRACT DEPLOYMENTS

The following addresses contain deployed to the Ethereum mainnet and verified smart contracts code that matches audited scope:

- UniswapV3: 0x457Ef4713933689D1FF13412DAC2683E4E8bb0A8
- UniswapV2: 0xEdBf8F73908c86a89f4D42344c8e01b82fE4Aaa6
- CurveV1: 0x49b34e58baB86B3cD23b0bE0aF4A152bE1056902
- YearnV2: 0x403E98b110a4DC89da963394dC8518b5f0E2D5fB
- AccountFactory: 0x444cd42baeddeb707eed823f7177b9abcc779c04
- AccountMining: 0x7B1AAF21AC0D420666B5966338FF9aEe763C29DF
- ACL: 0x523da3a8961e4dd4f6206dbf7e6c749f51796bb3
- AddressProvider: 0xcF64698AFF7E5f27A11dff868AF228653ba53be0
- ContractsRegister: 0xA50d4E7D8946a7c90652339CDBd262c375d54D99
- DataCompressor: 0x0050b1abd1dd2d9b01ce954e663ff3dbca9193b1
- WETHGateway: 0x4f952c4c5415b2609899abdc2f8f352f600d14d6
- CreditAccount: 0x373a292b93ff9017d28e64154ef83b99d5c4e270
- CreditFilter: 0x948d33a9537cf13bcc656218b385d19e5b6693e8
- CreditManager: 0x777e23a2acb2fcbb35f6ccf98272d03c722ba6eb
- LeveragedActions: 0xe11ac30edcfb16d0fcc2540d2c8253051ac93d49
- PriceOracle: 0x0e74a08443c5e39108520589176ac12ef65ab080
- YearnPriceFeed: 0x172971182351e00C2D700bA1e8c5586Ad2CFa38c
- LinearInterestRateModel: 0xf37d605f6428576529657e24dfb439803f602118
- PoolService: 0x24946bcbbd028d5abb62ad9b635eb1b1a67af668
- DieselToken: 0x6cfaf95457d7688022fc53e7abe052ef8dfbbdba
- GearToken: 0xba3335588d9403515223f109edc4eb7269a9ab5d
- Vesting: 0xEF686707193AF7406f40CBd7A39ba309Da5aD4Ec

2. FINDINGS REPORT

2.1 CRITICAL

CRT-1	Incorrect calculation of borrowed amount
File	CreditManager.sol
Severity	Critical
Status	Fixed at fa3265e0

DESCRIPTION

Total borrowed amount increases unequally, so total borrowed amount on credit accounts would be less than `totalBorrowed` on a `PoolService` which would lead to incorrect calcultaions for LP of a `PoolService`:
CreditManager.sol#L661

RECOMMENDATION

We recommend to change calculation of borrowed amount for credit accounts.

2.2 MAJOR

MJR-1	Possible remove of necessary adapter
File	CreditFilter.sol
Severity	Major
Status	Fixed at 0b825ffb

DESCRIPTION

In case 1 adapter is used for several contracts, then next lines will break work of these contracts:
CreditFilter.sol#L190
CreditFilter.sol#L217

RECOMMENDATION

We recommend adding a check, that removed adapter is not used in other contracts.

MJR-2	Incorrect change of state
File	CreditFilter.sol
Severity	Major
Status	Fixed at 0b825ffb

DESCRIPTION

Changing state to true is incorrect and must be done via `allowToken()`:
CreditFilter.sol#L398

RECOMMENDATION

We recommend allowing changing state only to false.

MJR-3	`creditManager` isn't checked
File	LeverageActions.sol
Severity	Major
Status	Fixed at 0b825ffb

DESCRIPTION

It is possible to give unlimited approve to poisoned contract:
LeverageActions.sol#L222

RECOMMENDATION

We recommend adding a check that `creditManager` is a system contract.

MJR-4	Uncounted fees in USDT
File	LeverageActions.sol
Severity	Major
Status	Fixed at 0b825ffb

DESCRIPTION

Contract can have less than `amountIn` because of fees in transferFrom function in USDT:
LeverageActions.sol#L232
LeverageActions.sol#L324

RECOMMENDATION

We recommend to call `openCreditAccount` with current contract balance.

MJR-5	Possible loss of assets by mistake
File	CreditManager.sol
Severity	Major
Status	Fixed at 0b825ffb

DESCRIPTION

If a liquidator calls this function by mistake with the next parameters: `to = address(0), force = true`, then he loses all assets:
CreditManager.sol#L302

RECOMMENDATION

We recommend adding a check that `to != address(0)`.

MJR-6	Broken account must be deleted
File	CreditManager.sol
Severity	Major
Status	Acknowledged

DESCRIPTION

In case if some tokens cannot be transferred from the account, then this account must be deleted:
CreditManager.sol#L451

RECOMMENDATION

We recommend to delete account in case some transfers are reverted.

MJR-7	Possible transfer of bad account
File	CreditManager.sol
Severity	Major
Status	Fixed at 0b825ffb

DESCRIPTION

User can have bad account with Hf < 1 but not liquidated yet, and transfer it to another user:
CreditManager.sol#L954

RECOMMENDATION

We recommend adding approve mechanic, so that account receiver does not receive account that he doesn't want.

MJR-8	Unnecessary allowance
File	YearnV2.sol
Severity	Major
Status	Fixed at 0b825ffb

DESCRIPTION

Vault's withdraw function burns shares, so allowance is unnecessary:
YearnV2.sol#L130

RECOMMENDATION

We recommend removing providing allowance.

MJR-9	Incorrect usage of function returned value
File	YearnV2.sol
Severity	Major
Status	Fixed at 0b825ffb

DESCRIPTION

Vault's withdraw function returns amount of tokens which was transferred, not shares: YearnV2.sol#L145

RECOMMENDATION

We recommend to change function code.

MJR-10	Incorrect taking out of `tail` account
File	AccountFactory.sol
Severity	Major
Status	Fixed at 0b825ffb

DESCRIPTION

If `creditAccount == tail` then `tail` is not updated properly and this will break the list of accounts:
AccountFactory.sol#L260

RECOMMENDATION

We recommend updating `tail` if it was taken out.

MJR-11	Incorrect minting
File	PoolService.sol
Severity	Major
Status	Fixed at 0b825ffb

DESCRIPTION

Because of fees in transferFrom function in USDT, contract would have less than `amount`: PoolService.sol#L155

RECOMMENDATION

We recommend to use balance difference to mint Diesel tokens.

MJR-12	Impossible liquidity removing
File	PoolService.sol
Severity	Major
Status	Acknowledged

DESCRIPTION

In case all funds were borrowed (big part of funds), users can't return their assets (`amountSent > balanceOf(address(this))`:
PoolService.sol#L182

RECOMMENDATION

We recommend adding a function for closing some account to return funds to LP.

MJR-13	Calculation can be incorrect
File	PoolService.sol
Severity	Major
Status	Acknowledged

DESCRIPTION

`expectedLiquidity` contains real balance + pseudo balance from borrowers interest: PoolService.sol#L396

RECOMMENDATION

We recommend adding a function for forcing closing some accounts to pay all balance for LP.

MJR-14	Impossible liquidation of broken account
File	CreditManager.sol
Severity	Major
Status	Fixed at 0b825ffb

DESCRIPTION

In case transfer was reverted, then specific token can't be accounted as collateral, and also liquidator must not pay the fee for this token:
CreditManager.sol#L593-L594

RECOMMENDATION

We recommend not to accumulate `tv` and `tvw` in case transfer was reverted.

MJR-15	Using tokens with whitelist function
File	CreditManager.sol
Severity	Major
Status	Fixed at 0b825ffb

DESCRIPTION

At line: CreditManager.sol#L584-L593

The Gearbox protocol is assumed to use some tokens which have whitelist function (ex. USDC, USDT) like a collateral. If this token will be blocked off-chain the Liquidates credit account function member of the `CreditManager` contract (CreditManager.sol#L300) will not work correctly because the internal function `_transferAssetsTo()` (CreditManager.sol#L584-L591) returns incorrect return value `totalValue`.

Example:

```
function _transferAssetsTo(
        address creditAccount,
        address to,
        bool force
    ) internal returns (uint256 totalValue, uint256 totalWeightedValue) {
        totalValue = 0;
        totalWeightedValue = 0;

        uint256 tokenMask;
        uint256 enabledTokens = creditFilter.enabledTokens(creditAccount);

        for (uint256 i = 0; i < creditFilter.allowedTokensCount(); i++) {
            tokenMask = 1 << i;
            if (enabledTokens & tokenMask > 0) {
                (
                    address token,
                    uint256 amount,
                    uint256 tv,
                    uint256 tvw
                ) = creditFilter.getCreditAccountTokenById(creditAccount, i);
                if (amount > 1) {
                    // The condition is met, but the transfer will not occur
                    // for blocked account
                    _safeTokenTransfer(
                        creditAccount,
                        token,
                        to,
                        amount.sub(1),
                        force
                    );
                    // In this case totalValue will not correct
                    totalValue += tv;
                    totalWeightedValue += tvw;
                }
```

```
                }
            }
```

RECOMMENDATION

Use solution where totalValue will not increase in case of unsuccessful transaction.

MJR-16	Looping a linked list
File	AccountFactory.sol
Severity	Major
Status	Fixed at ae615112

DESCRIPTION

At the line: AccountFactory.sol#L40
there's a linked list called `_nextCreditAccount`.
In this list, the account address refers to the address of the next account. Thus, a chain of addresses linked to each other is formed.
At the lines contracts/core/AccountFactory.sol#L248-L263 there's a `takeOut ()` function. It takes the credit account address from anywhere on the list and attaches it to the credit manager.
At the lines: AccountFactory.sol#L186-L199

there's a function `returnCreditAccount ()`. It returns the credit account address in the tail of the `_nextCreditAccount` linked list.
In the `takeOut()` function there is no logic for checking and zeroing the link to the next account from the taken account address.
When the taken address is returned to the tail of the list, it will contain the old value of the address. Loop through the list may occur when the function `_countCreditAccountsInStock ()` is used for the line: AccountFactory.sol#L352-L365.

RECOMMENDATION

It is necessary to zero the next item from the linked list after the AccountFactory.sol#L260 line:

```
_nextCreditAccount[creditAccount] = address(0);
```

MJR-17	No checking of element properties when returning it to the list
File	AccountFactory.sol
Severity	Major
Status	Fixed at 5e6a5d72

DESCRIPTION

At the lines: AccountFactory.sol#L186-L199
there's a function `returnCreditAccount ()`. It returns the credit account address in the tail of the `_nextCreditAccount` linked list.
For the line: AccountFactory.sol#L192,
the check is made that the value of `since ()` is not equal to `block.number`.
If the address has never been registered in an `AccountFactory` it will pass this check too.
But this address will not be added to the `creditAccounts` array and the logic of the entire contract will be violated.

RECOMMENDATION

It is necessary to make a code correction at the line contracts/core/AccountFactory.sol#L192:

```
ICreditAccount(usedAccount).since() != block.number &&
ICreditAccount(usedAccount).since() > 0,
```

2.3 WARNING

WRN-1	`priceFeeds` can't be changed
File	PriceOracle.sol
Severity	Warning
Status	Fixed at 0b825ffb

DESCRIPTION

`priceFeeds` can't be changed:
PriceOracle.sol#L56

RECOMMENDATION

We recommend adding a function for changing `priceFeeds`.

WRN-2	Work with incorrect decimals
File	PriceOracle.sol
Severity	Warning
Status	Fixed at 0b825ffb

DESCRIPTION

Price feed can return price with not 18 decimals:
PriceOracle.sol#L56

RECOMMENDATION

We recommend to check decimals of the `priceFeeds[token]`.

WRN-3	Unnecessary inheritance from `Proxy`
File	YearnPriceFeed.sol
Severity	Warning
Status	Fixed at **0b825ffb**

DESCRIPTION

All functions for `yVault` called from proxy wouldn't work because the storage is not the same:
YearnPriceFeed.sol#L17

RECOMMENDATION

We recommend removing inheritance from `Proxy`.

WRN-4	Incorrect input parameters
File	YearnPriceFeed.sol CreditFilter.sol LeverageActions.sol CreditManager.sol CurveV1.sol UniswapV2.sol UniswapV3.sol YearnV2.sol WETHGateway.sol AccountFactory.sol AccountMining.sol ACLTrait.sol DataCompressor.sol ContractsRegister.sol GearToken.sol Vesting.sol PoolService.sol
Severity	Warning
Status	Fixed at 0b825ffb

DESCRIPTION

Input data is not checked for possible containing of zero addresses:
YearnPriceFeed.sol#L24
CreditFilter.sol#L113
CreditFilter.sol#L597
LeverageActions.sol#L73
CreditManager.sol#L118
CurveV1.sol#L33
UniswapV2.sol#L28
UniswapV3.sol#L33
YearnV2.sol#L35
WETHGateway.sol#L85
WETHGateway.sol#L203
AccountFactory.sol#L298
AccountMining.sol#L27
ACLTrait.sol#L21
DataCompressor.sol#L58
ContractsRegister.sol#L36
ContractsRegister.sol#L66
GearToken.sol#L104
Vesting.sol#L43
PoolService.sol#L102

RECOMMENDATION

We recommend adding a check that input data is not equal to zero address.

WRN-5	Too many rights for configurator
File	CreditFilter.sol
Severity	Warning
Status	Acknowledged

DESCRIPTION

Configurator has too many rights and can steal all user's funds:
CreditFilter.sol

RECOMMENDATION

We recommend to use DAO for configurator.

WRN-6	Length of input arrays not checked
File	CreditFilter.sol
Severity	Warning
Status	Fixed at **0b825ffb**

DESCRIPTION

Length of input arrays is not checked:
CreditFilter.sol#L297

RECOMMENDATION

We recommend adding a check that lengths of arrays are equal.

WRN-7	`path` length not checked
File	LeverageActions.sol
Severity	Warning
Status	Acknowledged

DESCRIPTION

`path` length is not checked:
LeverageActions.sol#L107
LeverageActions.sol#L494

RECOMMENDATION

We recommend adding a check for `path` length.

CLIENT'S COMMENTARY

AUDITORS' COMMENT:
Not fixed, path.length must be checked

WRN-8	`wethGateway` can't be changed
File	CreditManager.sol
Severity	Warning
Status	Acknowledged

DESCRIPTION

`wethGateway` can't be changed, so in case of redeploy, all contracts must be redeployed too:
CreditManager.sol#L123

RECOMMENDATION

We recommend adding a function for updating `wethGateway`.

WRN-9	Account can be opened for zero address
File	CreditManager.sol
Severity	Warning
Status	Fixed at **0b825ffb**

DESCRIPTION

`onBehalfOf` can be equal to zero address:
CreditManager.sol#L180

RECOMMENDATION

We recommend adding a check that `onBehalfOf` is not equal to zero address.

WRN-10	Malicious user can pay less to pool
File	CreditManager.sol
Severity	Warning
Status	Acknowledged

DESCRIPTION

Malicious user can add poisoned pool to `paths[i]` to minimize profits and pay less to pool:
CreditManager.sol#L260

RECOMMENDATION

We recommend checking the total value of user before transfers.

CLIENT'S COMMENTARY

AUDITORS' COMMENT:
Not fixed, we recommend to add a whitelist for tokens in path.

WRN-11	Transfer to zero address
File	CreditManager.sol
Severity	Warning
Status	Fixed at 0b825ffb

DESCRIPTION

`to` can be equal to zero address:
CreditManager.sol#L338

RECOMMENDATION

We recommend adding a check that `to != address(0)`.

WRN-12	Possible transfer of 0 funds
File	CreditManager.sol
Severity	Warning
Status	Fixed at 0b825ffb

DESCRIPTION

For liquidation `remainingFunds` can be equal to zero:
CreditManager.sol#L434

RECOMMENDATION

We recommend adding a check that `remainingFunds != 0` before transfer.

WRN-13	Usage of ERC777 token can block liquidation
File	CreditManager.sol
Severity	Warning
Status	Acknowledged

DESCRIPTION

Malicious user can use ERC777 token callbacks for blocking liquidation:
CreditManager.sol#L433

RECOMMENDATION

We recommend not to use ERC777 token as underlying token.

WRN-14	Incorrect require
File	CreditManager.sol
Severity	Warning
Status	Fixed at 0b825ffb

DESCRIPTION

The following check is incorrect, because after this function actual borrowed amount of user would be = `borrowedAmount.add(timeDiscountedAmount)`:
CreditManager.sol#L649

RECOMMENDATION

We recommend adding `timeDiscountedAmount` instead of `amount`.

WRN-15	Possible incorrect setting value for `maxLeverageFactor`
File	CreditManager.sol
Severity	Warning
Status	Fixed at **0b825ffb**

DESCRIPTION

`maxLeverageFactor` can be set to 0:
CreditManager.sol#L722

RECOMMENDATION

We recommend adding a check that `_maxLeverageFactor > 0`.

WRN-16	User can approve any token
File	CreditManager.sol
Severity	Warning
Status	Fixed at 0b825ffb

DESCRIPTION

User can approve any token:
CreditManager.sol#L761

RECOMMENDATION

We recommend adding a check that user can only approve allowable tokens.

WRN-17	Incorrect length of input data
File	CreditManager.sol
Severity	Warning
Status	Fixed at 0b825ffb

DESCRIPTION

`paths.length` can be < `creditFilter.allowedTokensCount()`:
CreditManager.sol#L816

RECOMMENDATION

We recommend adding a check that arrays have equal lengths.

WRN-18	`amount` must be > 1
File	CreditManager.sol
Severity	Warning
Status	Fixed at 0b825ffb

DESCRIPTION

`amount` must be > 1:
CreditManager.sol#L821

RECOMMENDATION

We recommend changing the check.

WRN-19	`paths[i]` length not checked
File	CreditManager.sol
Severity	Warning
Status	Acknowledged

DESCRIPTION

`paths[i]` length is not checked, so underflow can occur:
CreditManager.sol#L828

RECOMMENDATION

We recommend checking `paths[i]` length.

CLIENT'S COMMENTARY

AUDITORS' COMMENT
Not fixed

WRN-20	Index not checked
File	CurveV1.sol
Severity	Warning
Status	Acknowledged

DESCRIPTION

Index `i` must be less than `N_COINS`:
CurveV1.sol#L40
CurveV1.sol#L58

RECOMMENDATION

We recommend adding a check that `i < N_COINS`.

WRN-21	Possible reentrancy
File	CurveV1.sol UniswapV2.sol UniswapV3.sol YearnV2.sol WETHGateway.sol PoolService.sol
Severity	Warning
Status	Fixed at 0b825ffb

DESCRIPTION

In the following function reentrancy can occur:
CurveV1.sol#L56
UniswapV2.sol#L55
UniswapV2.sol#L112
UniswapV3.sol#L49
UniswapV3.sol#L102
UniswapV3.sol#L152
UniswapV3.sol#L196
YearnV2.sol#L75
YearnV2.sol#L126
WETHGateway.sol#L107
WETHGateway.sol#L128
WETHGateway.sol#L159
WETHGateway.sol#L181
PoolService.sol#L242

RECOMMENDATION

We recommend to add a `nonReentrant` modificator.

WRN-22	Incorrect parameter passed
File	CurveV1.sol
Severity	Warning
Status	Fixed at 0b825ffb

DESCRIPTION

Curve swap can return more than `min_dy`:
CurveV1.sol#L82

RECOMMENDATION

We recommend to use balance difference as a parameter for the function.

WRN-23	Function doesn't exist
File	CurveV1.sol
Severity	Warning
Status	Fixed at 0b825ffb

DESCRIPTION

Function doesn't exist in Curve pool:
CurveV1.sol#L100
CurveV1.sol#L116

RECOMMENDATION

We recommend removing this function.

WRN-24	Incorrect function name
File	UniswapV2.sol
Severity	Warning
Status	Fixed at 0b825ffb

DESCRIPTION

Incorrect function of Uniswap-V2 pool is called:
UniswapV2.sol#L387

RECOMMENDATION

We recommend changing the function name.

WRN-25	`params.path` length not checked
File	UniswapV3.sol
Severity	Warning
Status	Acknowledged

DESCRIPTION

`params.path` length is not checked, so underflow can occur in `_extractTokens()` function:
UniswapV3.sol#L102
UniswapV3.sol#L200

RECOMMENDATION

We recommend adding a check for the `params.path` length.

WRN-26	Balance not checked
File	YearnV2.sol
Severity	Warning
Status	Acknowledged

DESCRIPTION

Balance of `creditAccount` is not checked:
YearnV2.sol#L83

RECOMMENDATION

We recommend adding a check that `balanceBefore > 0`.

WRN-27	Possible assets loss
File	WETHGateway.sol
Severity	Warning
Status	Fixed at 0b825ffb

DESCRIPTION

If `repayAmount > amount`, this means that Gateway repays credit from its own assets: WETHGateway.sol#L191

RECOMMENDATION

We recommend to call revert in case `repayAmount > amount`.

WRN-28	`masterCreditAccount` remains uninitialized
File	AccountFactory.sol
Severity	Warning
Status	Fixed at **0b825ffb**

DESCRIPTION

`masterCreditAccount` remains uninitialized, so anybody can initialize it:
AccountFactory.sol#L90

RECOMMENDATION

We recommend to initialize `masterCreditAccount`.

WRN-29	Account remains connected to previous credit manager
File	AccountFactory.sol
Severity	Warning
Status	Acknowledged

DESCRIPTION

After returning, credit account is still connected to previous credit manager: AccountFactory.sol#L196

RECOMMENDATION

We recommend connecting account to factory after returning.

WRN-30	Unnecessary list initialization
File	AccountFactory.sol
Severity	Warning
Status	Fixed at 0b825ffb

DESCRIPTION

Initialization of `_nextCreditAccount[tail]` if `tail == address(0)` is unnecessary: AccountFactory.sol#L241

RECOMMENDATION

We recommend adding a check for this.

WRN-31	`head` can't be taken out
File	AccountFactory.sol
Severity	Warning
Status	Fixed at 0b825ffb

DESCRIPTION

`prev` account for head doesn't exist:
AccountFactory.sol#L257

RECOMMENDATION

We recommend adding a special check for situation when `creditAccount == head`.

WRN-32	Incorrect update of list
File	AccountFactory.sol
Severity	Warning
Status	Fixed at 0b825ffb

DESCRIPTION

List isn't updated properly:
AccountFactory.sol#L260

RECOMMENDATION

We recommend to add `_nextCreditAccount[creditAccount] = address(0);`

WRN-33	Possible duplication of data
File	AccountFactory.sol
Severity	Warning
Status	Acknowledged

DESCRIPTION

If configurator calls function several times, then data duplication is possible: AccountFactory.sol#L301

RECOMMENDATION

We recommend adding a check that only new data is pushed to array.

WRN-34	`merkleProof` length not checked
File	AccountMining.sol
Severity	Warning
Status	Fixed at 0b825ffb

DESCRIPTION

`merkleProof` length is not checked:
AccountMining.sol#L64

RECOMMENDATION

We recommend adding a check that `merkleProof.length > 0`.

WRN-35	Unable to remove pool or manager
File	ContractsRegister.sol
Severity	Warning
Status	Acknowledged

DESCRIPTION

Pool or manager can't be removed:
ContractsRegister.sol#L38

RECOMMENDATION

We recommend adding functions for removing pools and managers.

WRN-36	`signatory` not checked
File	GearToken.sol
Severity	Warning
Status	Acknowledged

DESCRIPTION

`signatory` not checked:
GearToken.sol#L330

RECOMMENDATION

We recommend adding a check that correct `signatory` was recovered.

CLIENT'S COMMENTARY

AUDITORS' COMMENT:
Not fixed

WRN-37	`delegatee` not checked
File	GearToken.sol
Severity	Warning
Status	Acknowledged

DESCRIPTION

It is possible that `currentDelegate == delegatee`:
GearToken.sol#L407

RECOMMENDATION

We recommend adding a check that `currentDelegate != delegatee`.

WRN-38	`expectedLiquidityLimit` can be equal to zero
File	PoolService.sol
Severity	Warning
Status	Fixed at 0b825ffb

DESCRIPTION

`expectedLiquidityLimit` can be equal to zero:
PoolService.sol#L145

RECOMMENDATION

We recommend to set value to `expectedLiquidityLimit` in constructor.

WRN-39	Possible overflow can occur
File	PoolService.sol
Severity	Warning
Status	Fixed at 0b825ffb

DESCRIPTION

Possible overflow can occur:
PoolService.sol#L145

RECOMMENDATION

We recommend using safeMath for `expectedLiquidity() + amount`.

WRN-40	Transfer of 0 funds
File	PoolService.sol
Severity	Warning
Status	Fixed at 0b825ffb

DESCRIPTION

If `withdrawFee == 0`, then here 0 amount of asset transferred:
PoolService.sol#L191

RECOMMENDATION

We recommend adding a check, so 0 funds wouldn't be transferred.

WRN-41	`_timestampLU` can be equal to 0
File	PoolService.sol
Severity	Warning
Status	Acknowledged

DESCRIPTION

`_timestampLU` can be equal to 0, for example in constructor:
PoolService.sol#L209
PoolService.sol#L334

RECOMMENDATION

We recommend checking it and use another calculation for this scenario.

WRN-42	Forbidden manager never can use pool
File	PoolService.sol
Severity	Warning
Status	Acknowledged

DESCRIPTION

If a credit manager was forbidden, then he can't use the pool and nobody can change it:
PoolService.sol#L443

RECOMMENDATION

We recommend adding a function to allow usage of pool to forbidden managers.

WRN-43	Address not checked
File	PoolService.sol
Severity	Warning
Status	Fixed at 0b825ffb

DESCRIPTION

`_interestRateModel` can be equal to zero address:
PoolService.sol#L453

RECOMMENDATION

We recommend adding necessary checks for model address.

WRN-44	Possible overflow
File	CreditManager.sol
Severity	Warning
Status	Fixed at 0b825ffb

DESCRIPTION

If overall amount of tokens on credit account is very big, then overflow can occur: CreditManager.sol#L593-L594

RECOMMENDATION

We recommend to use safeMath.

WRN-45	User can't repay with force flag
File	CreditManager.sol
Severity	Warning
Status	Acknowledged

DESCRIPTION

If some token was blocked on account, user can't close this account and must wait until account would be liquidated by someone:
CreditManager.sol#L368

RECOMMENDATION

We recommend adding a parameter for the repay function, so that user can repay account with force flag.

WRN-46	Add condition
File	CreditFilter.sol
Severity	Warning
Status	Fixed at 5bfe544b

DESCRIPTION

For the line: CreditFilter.sol#L159
in the `allowToken ()` function, the existence of the `token` - `underlyingToken` pair is checked.
But it is better to immediately check if the value of this pair is greater than 0:

```
require(
IPriceOracle(priceOracle).getLastPrice(token, underlyingToken) > 0,
Errors.CF_TOKEN_IS_NOT_ALLOWED
);
```

RECOMMENDATION

It is recommended to make corrections to the source code.

WRN-47	Upgradeable `creditManager` params
File	CreditManager.sol
Severity	Warning
Status	Acknowledged

DESCRIPTION

Upgrade of `creditManager` params may lead to `creditAccount`s liquidation or break dependent protocols logic.
CreditManager.sol#L705

RECOMMENDATION

We recommend restricting params changing.

2.4 COMMENT

CMT-1	Unnecessary check
File	CreditFilter.sol
Severity	Comment
Status	Acknowledged

DESCRIPTION

This check is unnecessary if price is fetched from oracle:
CreditFilter.sol#L156

RECOMMENDATION

We recommend removing this check.

CMT-2	Unnecessary update
File	CreditFilter.sol
Severity	Comment
Status	Acknowledged

DESCRIPTION

If `contractToAdapter[targetContract] == adapter` , update is unnecessary:
CreditFilter.sol#L192

RECOMMENDATION

We recommend adding additional check.

CMT-3	Unnecessary initialization
File	CreditFilter.sol DataCompressor.sol GearToken.sol
Severity	Comment
Status	Fixed at 0b825ffb

DESCRIPTION

New `uint256` variables are equal to zero by default:
CreditFilter.sol#L297
DataCompressor.sol#L72
GearToken.sol#L107

RECOMMENDATION

We recommend removing initialization.

CMT-4	Unnecessary print to console
File	CreditFilter.sol
Severity	Comment
Status	Fixed at 0b825ffb

DESCRIPTION

Print to console can't be used in blockchain:
CreditFilter.sol#L343

RECOMMENDATION

We recommend removing print to console.

CMT-5	User can receive only ETH
File	CreditManager.sol
Severity	Comment
Status	Acknowledged

DESCRIPTION

User can receive only ETH:
CreditManager.sol#L626

RECOMMENDATION

We recommend adding a flag, so that users can choose asset for receiving (ETH or wETH).

CMT-6	Tokens can be locked on account
File	CreditAccount.sol
Severity	Comment
Status	Acknowledged

DESCRIPTION

Some tokens can be locked on account (for example USDT):
CreditAccount.sol

RECOMMENDATION

We recommend adding a function for transferring locked tokens to treasury, when account isn't used by manager.

CMT-7	Print to console
File	UniswapV3.sol
Severity	Comment
Status	Fixed at 0b825ffb

DESCRIPTION

This code was used for testing:
UniswapV3.sol#L230

RECOMMENDATION

We recommend removing these lines.

CMT-8	Unnecessary library for user types
File	Types.sol
Severity	Comment
Status	Acknowledged

DESCRIPTION

Unnecessary library for user types:
Types.sol

RECOMMENDATION

We recommend moving each struct to contract where it uses.

CMT-9	`wethAddress` can be const
File	WETHGateway.sol
Severity	Comment
Status	Acknowledged

DESCRIPTION

`wethAddress` can be const because it isn't changing:
WETHGateway.sol#L31

RECOMMENDATION

We recommend to set `wethAddress` as a constant.

CMT-10	Unnecessary safeMath
File	WETHGateway.sol
Severity	Comment
Status	Fixed at 0b825ffb

DESCRIPTION

safeMath here is unnecessary:
WETHGateway.sol#L192

RECOMMENDATION

We recommend not to use safeMath here.

CMT-11	Incorrect comment
File	AccountFactory.sol
Severity	Comment
Status	Fixed at 0b825ffb

DESCRIPTION

Comment for this variable is incorrect:
AccountFactory.sol#L54

RECOMMENDATION

We recommend changing the comment.

CMT-12	Similar functions are used
File	ACL.sol
Severity	Comment
Status	Acknowledged

DESCRIPTION

Functions with similar logic are used:
ACL.sol#L28

RECOMMENDATION

We recommend adding 1 function with bool parameter to set true/false.

CMT-13	`merkleRoot` can't be updated
File	AccountMining.sol
Severity	Comment
Status	Acknowledged

DESCRIPTION

`merkleRoot` can't be updated:
AccountMining.sol#L28

RECOMMENDATION

We recommend adding a function for updating `merkleRoot`.

CMT-14	All functions can be merged
File	AddressProvider.sol
Severity	Comment
Status	Acknowledged

DESCRIPTION

All functions can be merged into one with `bytes32` parameter:
AddressProvider.sol

RECOMMENDATION

We recommend merging all functions into 1 with `bytes32` parameter.

CMT-15	Visibility not set
File	ContractsRegister.sol
Severity	Comment
Status	Fixed at 0b825ffb

DESCRIPTION

Visibility is not set:
ContractsRegister.sol#L16
ContractsRegister.sol#L20

RECOMMENDATION

We recommend to set visibility.

CMT-16	Event not emitting
File	GearToken.sol
Severity	Comment
Status	Fixed at 0b825ffb

DESCRIPTION

After manager update, event is not emitting:
GearToken.sol#L115

RECOMMENDATION

We recommend to emit events after updating storage variables.

CMT-17	Range for variables not set
File	Vesting.sol
Severity	Comment
Status	Acknowledged

DESCRIPTION

Range for variables is not set:
Vesting.sol#L45

RECOMMENDATION

We recommend to add allowable range for variables

CMT-18	Two variables can be merged
File	Vesting.sol
Severity	Comment
Status	Acknowledged

DESCRIPTION

`started` can be merged with `cliffDuration`:
Vesting.sol#L46

RECOMMENDATION

We recommend merging `started` with `cliffDuration` into one variable.

CMT-19	Unnecessary setting on each mint
File	GearNFT.sol
Severity	Comment
Status	Fixed at 0b825ffb

DESCRIPTION

Setting base URI on each mint is unnecessary:
GearNFT.sol#L14

RECOMMENDATION

We recommend moving setting of base URI to constructor.

CMT-20	Unnecessary usage of variable
File	PoolService.sol
Severity	Comment
Status	Fixed at 0b825ffb

DESCRIPTION

`withdrawFee = 100 - withdrawMultiplier`, so it is unnecessary to store this variable: PoolService.sol#L191

RECOMMENDATION

We recommend removing `withdrawFee` and send `underlyingTokensAmount.sub(amountSent)`.

CMT-21	Parameters not checked
File	LinearInterestRateModel.sol
Severity	Comment
Status	Fixed at 0b825ffb

DESCRIPTION

Input parameters values are not checked:
LinearInterestRateModel.sol#L46

RECOMMENDATION

We recommend adding allowable range for each parameter.

CMT-22	The technical default of liquidity pool
File	CreditAccount.sol
Severity	Comment
Status	Acknowledged

DESCRIPTION

In some cases, the liquidity providers can do a mass withdraw requests. It may happen on pending protocol changes and so on. However, liquidity can not be withdrawn while it is in use by a credit account. In such case, some liquidity providers may be unable to withdraw their liquidity until credit account releases the liquidity.

RECOMMENDATION

We recommend implementing a forced liquidation of credit accounts by withdraw request. Certainly, the liquidator should be fined and the borrower should be rewarded on such liquidation.

CMT-23	Undesired side effects of address reusing
File	CreditAccount.sol
Severity	Comment
Status	Acknowledged

DESCRIPTION

A credit account is reusing the same ethereum address for different borrowers. If this address is punished for some reason during activity of one borrower, the other borrower may suffer undesired side effects like having bad debt in protocols like Compound and so on.

RECOMMENDATION

We recommend to abandon address reusing. One borrower - one ethereum address.

3. ABOUT MIXBYTES

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

BLOCKCHAINS

- Ethereum
- Cosmos
- EOS
- Substrate

TECH STACK

- Python
- Solidity
- Rust
- C++

CONTACTS

https://github.com/mixbytes/audits_public

https://mixbytes.io/

hello@mixbytes.io

https://t.me/MixBytes

https://twitter.com/mixbytes