Master Protocol
Audit Report

Tue Jun 18 2024

contact@bitslab.xyz https://twitter.com/scalebit_

ScaleBit

Master Protocol Audit Report

1 Executive Summary

1.1 Project Information

Description	Groundbreaking New Decentralized Financial Protocol
Type	DeFi
Auditors	ScaleBit
Timeline	Tue Jun 04 2024 - Mon Jun 17 2024
Languages	Solidity
Platform	BTC
Methods	Architecture Review, Unit Testing, Manual Review

1.2 Files in Scope

The following are the SHA1 hashes of the original reviewed files.

ID	File	SHA-1 Hash
BSCFC	master-protocol-contract-470951e657b0/contracts/offchain-helpers/BaseSplitCodeFactoryContract.sol	03a6bb0bf7cc107ed6fc5143d2e96ac75dbefb7a
MGS	master-protocol-contract-470951e657b0/contracts/offchain-helpers/router-static/MasterGatewayStatic.sol	e0efdcdaab36e109c897b223b98ba7477ceae243
SLA	master-protocol-contract-470951e657b0/contracts/offchain-helpers/router-static/base/StorageLayout.sol	dd6376ae33374858ca996be8bba56a2b14e4b8b9
ASS	master-protocol-contract-470951e657b0/contracts/offchain-helpers/router-static/base/ActionStorageStatic.sol	accd4e0b353b365b3501975533137d331727dda2
AMAS	master-protocol-contract-470951e657b0/contracts/offchain-helpers/router-static/base/ActionMarketAuxStatic.sol	93356628128a4372a7862be3905169aab9819c27
AMRS	master-protocol-contract-470951e657b0/contracts/offchain-helpers/router-static/base/ActionMintRedeemStatic.sol	1c170fa9032064538268347a77cd81232ef619ce
AIS	master-protocol-contract-470951e657b0/contracts/offchain-helpers/router-static/base/ActionInfoStatic.sol	d2d122ab3d09f1383e019fdd07d290df3746482a

AMCS	master-protocol-contract-470951e657b0/contracts/offchain-helpers/router-static/base/ActionMarketCoreStatic.sol	d3843fc614ff756e87d1d218228d64ee51f27f9d
IAMCS	master-protocol-contract-470951e657b0/contracts/interfaces/IActionMarketCoreStatic.sol	92aa2dee8fd87fb38df6bc92aab466417b2b83b0
IMF	master-protocol-contract-470951e657b0/contracts/interfaces/IMarketFactory.sol	ef9621eb8f0876830408398f7f615666ecd8f770
IIMYT	master-protocol-contract-470951e657b0/contracts/interfaces/IInterestManagerYT.sol	3d2a483d1cad33d0fe059235d877af6dbdb6afb9
IAARL	master-protocol-contract-470951e657b0/contracts/interfaces/IActionAddRemoveLiq.sol	b8f5a96cae7b48a6008469e1781adb16666b81ac
IPT	master-protocol-contract-470951e657b0/contracts/interfaces/IPrincipalToken.sol	f19e381b8dd708a245cbb82bdfbe5207f5857352
IAC	master-protocol-contract-470951e657b0/contracts/interfaces/IActionCallback.sol	ecb407505de494187f33ce44d74db89e3c23da4a
ILR	master-protocol-contract-470951e657b0/contracts/interfaces/ILimitRouter.sol	66b6ffd74990f57902f12c9d1bf41a3378a75a0d
IMA	master-protocol-contract-470951e657b0/contracts/interfaces/IMarket.sol	cc3bed543769b70f595fc5235a9a94b0947cbe86
IMSC	master-protocol-contract-470951e657b0/contracts/interfaces/IMarket	e7387f3e7f215a6a0b57fda7b838d3c4c8b95af0

	SwapCallback.sol	
IAS	master-protocol-contract-470951e657b0/contracts/interfaces/IActionStorage.sol	07d3cb303fc150fb11f825708bdd26cfffde96c2
IGC	master-protocol-contract-470951e657b0/contracts/interfaces/IGaugeController.sol	2938a08b223f50902ac95ee0c45275d64dbf3570
IAMRS	master-protocol-contract-470951e657b0/contracts/interfaces/IActionMintRedeemStatic.sol	f37be22a89d8eb4bc01341a2b1e9e0e5f2e5a2c3
IRS	master-protocol-contract-470951e657b0/contracts/interfaces/IRouterStatic.sol	0c07a5404f4e4c6b038bf789774537c3b3f3ae77
IRM	master-protocol-contract-470951e657b0/contracts/interfaces/IRewardManager.sol	839f26167294f0395c88fb74c506e1b9f58ae8e7
IMPBTC	master-protocol-contract-470951e657b0/contracts/interfaces/IMPBTC.sol	99b02ec14ba47d86941243d031c8c116b04fc9b5
IPVT	master-protocol-contract-470951e657b0/contracts/interfaces/IPVeToken.sol	d5c1d45986ed9db53a75ce8e1112acbafe068958
IYCF	master-protocol-contract-470951e657b0/contracts/interfaces/IYieldContractFactory.sol	5fbb9711752ee8397056eb511aa3bc8495f60820
IASPT	master-protocol-contract-470951e657b0/contracts/interfaces/IActionSwapPT.sol	34c5ee832caee1fb462d4be7190d1221e9b51735
IYT	master-protocol-contract-470951e657b0/contracts/interfaces/IYieldT	f829d60b007c93ce3c611d732709ca38eefa21f4

	oken.sol	
IAMAS	master-protocol-contract-470951e657b0/contracts/interfaces/IActionMarketAuxStatic.sol	1557ddc437fb6532646d66898908cfe028bd5d1e
IASYT	master-protocol-contract-470951e657b0/contracts/interfaces/IActionSwapYT.sol	9e9b549ee3540296158cd070a11b21296e76f4bb
IASS	master-protocol-contract-470951e657b0/contracts/interfaces/IActionStorageStatic.sol	9ab7375ffcb4c70951033ed50a20b578b7b5ebbc
IAAT	master-protocol-contract-470951e657b0/contracts/interfaces/IAllActionType.sol	6fcdf9703bb1f268ba8b6cf775e7ee6c551a2375
IGA	master-protocol-contract-470951e657b0/contracts/interfaces/IGauge.sol	3f38db890de3a865bb5cf5f7c6a15965cda32cbf
IAVMS	master-protocol-contract-470951e657b0/contracts/interfaces/IActionVeMasterStatic.sol	9e35506ea2c45618fc51c3aa52e16dd379e5f3ee
IAIS	master-protocol-contract-470951e657b0/contracts/interfaces/IActionInfoStatic.sol	ded8371d9537b99a82114601413d92e8064e8782
IMD	master-protocol-contract-470951e657b0/contracts/interfaces/IMiniDiamond.sol	2e3e043ee264a3ab7c85a1f3be45f07d60bab1b8
IAM	master-protocol-contract-470951e657b0/contracts/interfaces/IActionMisc.sol	d6c22331d644885a6b11d8de02fc75aac1abbf1e
ISY	master-protocol-contract-470951e657b0/contracts/interfaces/IStanda	b1d62982b2c8358e1d05098b81752f26acec32bd

	rdizedYield.sol	
ILD	master-protocol-contract-470951e657b0/contracts/interfaces/ILinearDistributor.sol	a0c1e15c315484a7821ec3bbd6b3666c07989825
SLI	master-protocol-contract-470951e657b0/contracts/core/libraries/StringLib.sol	1f5b76eadce29e906ef94b91d82991a70713788d
THE	master-protocol-contract-470951e657b0/contracts/core/libraries/TokenHelper.sol	5fb37b7c1ca2af9d7fa828f549da08830493e8a5
ALI	master-protocol-contract-470951e657b0/contracts/core/libraries/ArrayLib.sol	59e5cf84342e855c2ef1a4e92df1798d755a3a90
EUL	master-protocol-contract-470951e657b0/contracts/core/libraries/ExpiryUtilsLib.sol	fc2b3785399c7fbf539ae6a8b9a9d4fcf9057303
MHE	master-protocol-contract-470951e657b0/contracts/core/libraries/MiniHelpers.sol	d5b894bc123ca16cd6d4ec715656f13539527a67
ERR	master-protocol-contract-470951e657b0/contracts/core/libraries/Errors.sol	5b05ee5ccc2da3cb61b92a693d29c05ba0414972
BOU	master-protocol-contract-470951e657b0/contracts/core/libraries/BoringOwnableUpgradeable.sol	1d79dbc650ca4fe7400bdda262e7cd239dfdc7fd
BSCF	master-protocol-contract-470951e657b0/contracts/core/libraries/BaseSplitCodeFactory.sol	cba3193a7fee409a30d273560458796a1a29f4f4
PMA	master-protocol-contract-470951e657b0/contracts/core/libraries/mat	2fb2f8323f8a59b7e3f2fe5f12c505364f824676

	h/PMath.sol	
LEM	master-protocol-contract-470951e657b0/contracts/core/libraries/math/LogExpMath.sol	f119fb9aedf5f618aadef4c028a4cf47ce78d08f
MMC	master-protocol-contract-470951e657b0/contracts/core/market/MarketMathCore.sol	335cee0b1eddca575db001f00a59aab75d64daef
OLI	master-protocol-contract-470951e657b0/contracts/core/market/OracleLib.sol	631479c5b9dd54adc28027e5e5e157e64ef0235d
MMA	master-protocol-contract-470951e657b0/contracts/core/market/MasterMarket.sol	135fa4baeda9b9924321e9fbc6aeb43d67644532
MMF	master-protocol-contract-470951e657b0/contracts/core/market/MasterMarketFactory.sol	997e46a92fe22ae16f429d768e55ead1034d7f70
MGA	master-protocol-contract-470951e657b0/contracts/core/market/MasterGauge.sol	e6aa58cea7dab7e4f8c9d2ca69942ffd31df400f
RMA	master-protocol-contract-470951e657b0/contracts/core/reward-manager/RewardManagerAbstract.sol	743d163b476350db15fcc584f7f93a0856f9dd38
MPC4E6B0CCRMRMS	master-protocol-contract-470951e657b0/contracts/core/reward-manager/RewardManager.sol	e828baa6ce66e3bfadbd82fee2d8124c54ef5e4e
MPT	master-protocol-contract-470951e657b0/contracts/core/py/MasterPrincipalToken.sol	2fe5389b40e4e3c5af613988372ec6d38028aa1c
MYT	master-protocol-contract-470951e657b0/contracts/core/py/MasterYie	fade90de235622762473976cec908a1db789a2bf

	ldToken.sol	
IMYT	master-protocol-contract-470951e657b0/contracts/core/py/InterestManagerYT.sol	1a0e5ef9f7e5803c32baf5284ccce25ebb490761
MYCF	master-protocol-contract-470951e657b0/contracts/core/py/MasterYieldContractFactory.sol	6fe1a77a1272e921ad54034588fe8ce81ea99c35
SYB	master-protocol-contract-470951e657b0/contracts/core/sy/SYBase.sol	4c47d1131ed205103dec948256f1d64e1f16a4ce
MMBTCSY	master-protocol-contract-470951e657b0/contracts/core/sy/implementations/MasterMpBTCSY.sol	4a7d0b4bb2dde277bc1f1794e7d924c93a72e79e
ISBB	master-protocol-contract-470951e657b0/contracts/core/sy/implementations/bouncebit/IStBB.sol	e0867790ad891796719eb24da814904b679cc465
MSBBSY	master-protocol-contract-470951e657b0/contracts/core/sy/implementations/bouncebit/MasterStBBSY.sol	8b47c072f8e4573275c86d800e62fcb655311ba1
ISBBTC	master-protocol-contract-470951e657b0/contracts/core/sy/implementations/bouncebit/IStBBTC.sol	1a831c0c7fd7773e71d48e3e36bd6025c4491bdd
MSBBTCSY	master-protocol-contract-470951e657b0/contracts/core/sy/implementations/bouncebit/MasterStBBTCSY.sol	784d8c3d14f3342e37bf89fedc084d29b5c5db27
SYBWR	master-protocol-contract-470951e657b0/contracts/core/sy/SYBaseWithRewards.sol	975f3c191a7a511073a97569250fb9b4c378f7f1

SYU	master-protocol-contract-470951e657b0/contracts/core/sy/SYUtils.sol	4f07b796bdcb46f25a2f7525aa0fcbb11766b3f7
PYI	master-protocol-contract-470951e657b0/contracts/core/sy/PYIndex.sol	0cef00c3559880bef42f50cae7e02fa70b79ccbb
MERC2P	master-protocol-contract-470951e657b0/contracts/core/erc20/MasterERC20Permit.sol	109dc40ed1190ba560efe12aea033ad2b4827de6
MERC2	master-protocol-contract-470951e657b0/contracts/core/erc20/MasterERC20.sol	9562fa71396d83bca2b63ed3470d8e2ec68d7785
MERC2U	master-protocol-contract-470951e657b0/contracts/core/erc20/MasterERC20Upg.sol	4127e5dd1aaf17069176464fd578b325e625bdce
MERC2PU	master-protocol-contract-470951e657b0/contracts/core/erc20/MasterERC20PermitUpg.sol	4a2739c3fc9fb15968ee2a2408a3ea32322cfd4f
AST	master-protocol-contract-470951e657b0/contracts/gateway/ActionStorage.sol	696b6ba748afb993531482e34d421c000eb95302
AARL	master-protocol-contract-470951e657b0/contracts/gateway/ActionAddRemoveLiq.sol	58f9ac26b53108f33daba419a58c91cb1ae2806a
ISA	master-protocol-contract-470951e657b0/contracts/gateway/swap-aggregator/ISwapAggregator.sol	ce58cd8f24903408f6d1bc98e7d071a830bb67ef
ASPT	master-protocol-contract-470951e657b0/contracts/gateway/ActionSwapPT.sol	7037221fe9cc82b337e2be5ab34e26d279a45f0b

ASYT	master-protocol-contract-470951e657b0/contracts/gateway/ActionSwapYT.sol	01c8139873cbd7ac7f79ded2ba6cab119638a719
ACA	master-protocol-contract-470951e657b0/contracts/gateway/ActionCallback.sol	0188bbe3793e404484d959ba6cab94942bc528d1
MPC4E6B0CGMGS	master-protocol-contract-470951e657b0/contracts/gateway/MasterGateway.sol	d7bcd0c18d81d32e8aa2d9d07f618baded5cc240
MAL	master-protocol-contract-470951e657b0/contracts/gateway/base/MarketApproxLib.sol	e0da6165d535a067ddc6719138f37094222092b9
ABA	master-protocol-contract-470951e657b0/contracts/gateway/base/ActionBase.sol	234d337b41bb4a46c3f246fb8bd923e0be7a9251
CHE	master-protocol-contract-470951e657b0/contracts/gateway/base/CallbackHelper.sol	32342c9e411fc0fc48d499d52d2346e64757cdf3
GST	master-protocol-contract-470951e657b0/contracts/gateway/GatewayStorage.sol	ee624b185479340d0d70b07e55d69fd8209b26c6
AMI	master-protocol-contract-470951e657b0/contracts/gateway/ActionMisc.sol	93c8a707f4cc7f0252ce20a74d5b97107341372b

1.3 Issue Statistic

Item	Count	Fixed	Acknowledged
Total	8	1	7
Informational	5	1	4
Minor	3	0	3
Medium	0	0	0
Major	0	0	0
Critical	0	0	0

1.4 ScaleBit Audit Breakdown

ScaleBit aims to assess repositories for security-related issues, code quality, and compliance with specifications and best practices. Possible issues our team looked for included (but are not limited to):

- Transaction-ordering dependence
- Timestamp dependence
- Integer overflow/underflow
- Number of rounding errors
- Unchecked External Call
- Unchecked CALL Return Values
- Functionality Checks
- Reentrancy
- Denial of service / logical oversights
- Access control
- Centralization of power
- Business logic issues
- Gas usage
- Fallback function usage
- tx.origin authentication
- Replay attacks
- Coding style issues

1.5 Methodology

The security team adopted the **"Testing and Automated Analysis"**, **"Code Review"** and **"Formal Verification"** strategy to perform a complete security test on the code in a way that is closest to the real attack. The main entrance and scope of security testing are stated in the conventions in the "Audit Objective", which can expand to contexts beyond the scope according to the actual testing needs. The main types of this security audit include:

(1) Testing and Automated Analysis

Items to check: state consistency / failure rollback / unit testing / value overflows / parameter verification / unhandled errors / boundary checking / coding specifications.

(2) Code Review

The code scope is illustrated in section 1.2.

(3) Audit Process

- Carry out relevant security tests on the testnet or the mainnet;
- If there are any questions during the audit process, communicate with the code owner in time. The code owners should actively cooperate (this might include providing the latest stable source code, relevant deployment scripts or methods, transaction signature scripts, exchange docking schemes, etc.);
- The necessary information during the audit process will be well documented for both the audit team and the code owner in a timely manner.

2 Summary

This report has been commissioned by Master Protocol to identify any potential issues and vulnerabilities in the source code of the Master Protocol smart contract, as well as any contract dependencies that were not part of an officially recognized library. In this audit, we have utilized various techniques, including manual code review and static analysis, to identify potential vulnerabilities and security issues.

During the audit, we identified 8 issues of varying severity, listed below.

ID	Title	Severity	Status
AMI-1	Redundant Code	Informational	Fixed
EUL-1	Packing Time Sequence	Informational	Acknowledged
MSB-1	The Reward Balance Failed To Update And Cannot Be Claimed	Informational	Acknowledged
MYT-1	`_pyIndexCurrent` Maybe Affects ExchangeRate	Informational	Acknowledged
RMA-1	RewardToken Cannot Be Underlying Yield Token	Minor	Acknowledged
MMA1-1	The Initial Liquidity Provider Will Lose Some LP	Minor	Acknowledged
MMB1-1	External Function Dependencies	Minor	Acknowledged
SYB1-1	Protocol Does Not Support Deflationary Tokens	Informational	Acknowledged

3 Participant Process

Here are the relevant actors with their respective abilities within the Master Protocol Smart Contract :

Audit Scope

- Master Protocol is a fork project with some innovative improvements, based on commit hash 5459a015ed37c7caa1da4403e226dc29e3e39318.
- The audit scope mainly includes effective code changes except for name changes and name interface adaptation.
- Some contracts use the proxy mode, and the specific implementation contracts are not within the scope of the audit.

Users

- The project party will initialize the yieldToken in the Master protocol, and the user will use the MasterMpBTCSY contract to encapsulate the basic token into SY Token.
- Users can obtain SY token certificates by staking `yieldToken` (base tokens) through the `deposit` function.
- Users can redeem SY tokens by burning them to redeem `yieldToken` (base tokens).
- The `MasterstBBSY` contract provides users with a way to package tokens with the platform currency SY, and when redeeming tokens, they receive the corresponding yieldToken.
- The MasterStBBTCSY contract uses a fixed `exchangeRate` , and users can get `curenncy BB` rewards.
- Users can use `mint` to add liquidity in the master protocol to obtain the corresponding LP tokens.
- Users can use `burn` to remove liquidity to obtain `SY` and `PT` .
- Users can exchange tokens in the master protocol, such as `PT` , `SY` , `YT` .

4 Findings

AMI-1 Redundant Code

Severity: Informational

Status: Fixed

Code Location:

master-protocol-contract-470951e657b0/contracts/gateway/ActionMisc.sol#143

Descriptions:

1. The `actionRegisterTest` function returns a fixed bool value of `true` and is not used in other contracts. It only adds an interface and has no clear purpose in the contract.
2. Invalid test code, and console.log.
3. Uncomment the Todo code in the MasterGauge.sol code to check if it should be executed.

```
constructor(address _SY, address _veMaster, address _gaugeController) {
    SY = _SY;
    // tdo: uncomment this
    // veMASTER = IPVeToken(_veMaster);
    // gaugeController = _gaugeController;
    // MASTER = IGaugeController(gaugeController).master();
}
```

Suggestion:

It is recommended to confirm the business logic.

Resolution:

The log function and test function have been deleted.

EUL-1 Packing Time Sequence

Severity: Informational

Status: Acknowledged

Code Location:

master-protocol-contract-470951e657b0/contracts/core/libraries/ExpiryUtilsLib.sol#47

Descriptions:

The `toFormatDate` function is designed with reference to `toRFC2822String`. It is packaged in the opposite direction. When parsing, you may need to handle the order in which related functions in the contract are read separately.

```
s = string(abi.encodePacked(year,month,day));
```

Suggestion:

It is recommended to package according to the same design direction.

MSB-1 The Reward Balance Failed To Update And Cannot Be Claimed

Severity: Informational

Status: Acknowledged

Code Location:

master-protocol-contract-470951e657b0/contracts/core/sy/implementations/bouncebit/MasterStBBTCSY.sol#78

Descriptions:

When the current block triggers `_updateRewardIndex` to update lastBalance, rewards are collected through `stBBTC`. If the contract balance is insufficient after the previous block issued rewards and the rewards collected in the current block do not increase linearly, and are less than the last updated lastBalance, the update will fail. This will cause the dependent `claimRewards` function acc to fail to update and cannot collect rewards. The only way is to wait until the block reward is greater than lastBalance.

```
function _redeemExternalReward() internal override {
    IStBBTC(stBBTC).getReward();
}


function _updateRewardIndex()
    internal
    virtual
    override
    returns (address[] memory tokens, uint256[] memory indexes)
{
    tokens = _getRewardTokens();
    indexes = new uint256[](tokens.length);

    if (tokens.length == 0) return (tokens, indexes);

    if (lastRewardBlock != block.number) {
        // if we have not yet update the index for this block
        lastRewardBlock = block.number;
```

```
        uint256 totalShares = _rewardSharesTotal();

        _redeemExternalReward();

        for (uint256 i = 0; i < tokens.length; ++i) {
            address token = tokens[i];

            // the entire token balance of the contract must be the rewards of the contract

            RewardState memory _state = rewardState[token];
            (uint256 lastBalance, uint256 index) = (_state.lastBalance, _state.index);

            uint256 accrued = _selfBalance(tokens[i]) - lastBalance;

            if (index == 0) index = INITIAL_REWARD_INDEX;
            if (totalShares != 0) index += accrued.divDown(totalShares);

            rewardState[token] = RewardState({
                index: index.Uint128(),
                lastBalance: (lastBalance + accrued).Uint128()
            });
            indexes[i] = index;
        }
    } else {
        for (uint256 i = 0; i < tokens.length; i++) {
            indexes[i] = rewardState[tokens[i]].index;
        }
    }
}
```

Suggestion:

It is recommended that when designing `stBBTC`, the `getReward` function checks the amount of rewards to be issued in the master contract to ensure that the update is complete. For example, if `stBBTC` has a withdraw function to extract native curenncy BB, the reward issuance part will be retained.

MYT-1 `_pyIndexCurrent` Maybe Affects ExchangeRate

Severity: Informational

Status: Acknowledged

Code Location:

master-protocol-contract-470951e657b0/contracts/core/py/MasterYieldToken.sol#247,337

Descriptions:

The `_pyIndexCurrent` function will take the historical highest exchange rate, and the PY token and PT AMM markets can still continue to operate fairly. If exchangeRate() is manipulable, exchangeRate is reading the current balance of the underlying token as part of totalAsset or totalValue divided by totalSupply or totalShares, and the mint() function somehow allows external calls such as "flashmint, swapmint" after the token is transferred in and before the new shares are minted, the attacker can call `_pyIndexCurrent` in MasterYieldToken.sol to update to a temporarily inflated exchange rate.

```
  function _pyIndexCurrent() internal returns (uint256 currentIndex) {
    if (doCacheIndexSameBlock && pyIndexLastUpdatedBlock == block.number) return
_pyIndexStored;

    uint128 index128 = PMath.max(IStandardizedYield(SY).exchangeRate(),
_pyIndexStored).Uint128();

    currentIndex = index128;
    _pyIndexStored = index128;
    pyIndexLastUpdatedBlock = uint128(block.number);
  }
```

Suggestion:

It is recommended not to use the real-time exchangeRate when designing sytoken.

RMA-1 RewardToken Cannot Be Underlying Yield Token

Severity: Minor

Status: Acknowledged

Code Location:

master-protocol-contract-470951e657b0/contracts/core/reward-manager/RewardManager.sol#15

Descriptions:

In `Master:RewardManager:_updateRewardIndex` , if the newly added rewardToken happens to be the underlying revenue token, the contract will mistakenly treat the balance of all underlying revenue tokens as the new reward.

```
function _updateRewardIndex()
    internal
    virtual
    override
    returns (address[] memory tokens, uint256[] memory indexes)
{
    tokens = _getRewardTokens();
    indexes = new uint256[](tokens.length);

    if (tokens.length == 0) return (tokens, indexes);

    if (lastRewardBlock != block.number) {
        // if we have not yet update the index for this block
        lastRewardBlock = block.number;

        uint256 totalShares = _rewardSharesTotal();

        _redeemExternalReward();

        for (uint256 i = 0; i < tokens.length; ++i) {
            address token = tokens[i];

            // the entire token balance of the contract must be the rewards of the contract

            RewardState memory _state = rewardState[token];
            (uint256 lastBalance, uint256 index) = (_state.lastBalance, _state.index);
```

```
            uint256 accrued = _selfBalance(tokens[i]) - lastBalance;

            if (index == 0) index = INITIAL_REWARD_INDEX;
            if (totalShares != 0) index += accrued.divDown(totalShares);

            rewardState[token] = RewardState({
                index: index.Uint128(),
                lastBalance: (lastBalance + accrued).Uint128()
            });
            indexes[i] = index;
        }
    } else {
        for (uint256 i = 0; i < tokens.length; i++) {
            indexes[i] = rewardState[tokens[i]].index;
        }
    }
}

function _selfBalance(address token) internal view returns (uint256) {
    return (token == NATIVE) ? address(this).balance :
IERC20(token).balanceOf(address(this));
}
```

Suggestion:

It is recommended not to use `yieldToken` as `rewardToken`.

MMA1-1 The Initial Liquidity Provider Will Lose Some LP

Severity: Minor

Status: Acknowledged

Code Location:

master-protocol-contract-470951e657b0/contracts/core/market/MasterMarket.sol#111

Descriptions:

The current implementation will permanently lock a certain amount of LP tokens for the initial liquidity providers.

```
    function mint(
        address receiver,
        uint256 netSyDesired,
        uint256 netPtDesired
    ) external nonReentrant notExpired returns (uint256 netLpOut, uint256 netSyUsed,
uint256 netPtUsed) {
...

        // initializing the market
        if (lpToReserve != 0) {
            market.setInitialLnImpliedRate(index, initialAnchor, block.timestamp);
            _mint(address(1), lpToReserve);
        }

        _mint(receiver, netLpOut);

        _writeState(market);

...

        emit Mint(receiver, netLpOut, netSyUsed, netPtUsed);
    }
```

Suggestion:

It is recommended to consider storing the address of the initial liquidity provider and allow the initial liquidity provider to _burn(address(1), lpToReserve); after expiration.

MMB1-1 External Function Dependencies

Severity: Minor

Status: Acknowledged

Code Location:

master-protocol-contract-470951e657b0/contracts/core/sy/implementations/MasterMpBTCSY.sol#18,26;

master-protocol-contract-470951e657b0/contracts/core/sy/implementations/bouncebit/MasterStBBSY.sol#43

Descriptions:

1. The external token interface `getMPBtcByShares` needs to carry precision when processing the return value, because the precision of 1e18 is handled in the MasterMpBTCSY contract.
2. Avoid using address(this) to obtain the real-time exchange rate in the implementation of the `getMPBtcByShares/getPooledNativeByShares` function, which may lead to price issues such as read-only reentry and incorrect exchange ratios during redeem/deposit.
3. Ensure the return value rate is accurate and not 0.

```
amountSharesOut = (amountDeposited * 1e18) / exchangeRate();
rate = IMPBTC(yieldToken).getMPBtcByShares(1e18);
```

```
  function exchangeRate() public view virtual override returns (uint256 rate) {
    rate = IStBB(yieldToken).getPooledNativeByShares(1e18);
  }
```

Suggestion:

It is recommended to set the query function `getMPBtcByShares/getPooledNativeByShares` to carry precision when implementing YT token and avoid extreme values such as close to 0, which will cause serious precision loss after SY token packaging.

Resolution:

The client has identified and understood the external risk causes.

SYB1-1 Protocol Does Not Support Deflationary Tokens

Severity: Informational

Status: Acknowledged

Code Location:

master-protocol-contract-470951e657b0/contracts/core/sy/SYBase.sol#47

Descriptions:

Some ERC20 tokens have modifications to their ERC20 transfer or balanceOf functions. One of these tokens is a deflationary token that charges a fee for each transfer() or transferFrom(). In the SY.deposit function, the prepaid fee amount `_transferIn` is used to calculate the shares of the base token `amountTokenToDeposit` instead of the actual amount received. This may result in receiving the wrong number of shares compared to deposits using other base tokens.

Suggestion:

It is recommended to be aware of this limitation when using transfer fee tokens as base tokens.

Appendix 1

Issue Level

- **Informational** issues are often recommendations to improve the style of the code or to optimize code that does not affect the overall functionality.
- **Minor** issues are general suggestions relevant to best practices and readability. They don't post any direct risk. Developers are encouraged to fix them.
- **Medium** issues are non-exploitable problems and not security vulnerabilities. They should be fixed unless there is a specific reason not to.
- **Major** issues are security vulnerabilities. They put a portion of users' sensitive information at risk, and often are not directly exploitable. All major issues should be fixed.
- **Critical** issues are directly exploitable security vulnerabilities. They put users' sensitive information at risk. All critical issues should be fixed.

Issue Status

- **Fixed:** The issue has been resolved.
- **Partially Fixed:** The issue has been partially resolved.
- **Acknowledged:** The issue has been acknowledged by the code owner, and the code owner confirms it's as designed, and decides to keep it.

Appendix 2

Disclaimer

This report is based on the scope of materials and documents provided, with a limited review at the time provided. Results may not be complete and do not include all vulnerabilities. The review and this report are provided on an as-is, where-is, and as-available basis. You agree that your access and/or use, including but not limited to any associated services, products, protocols, platforms, content, and materials, will be at your own risk. A report does not imply an endorsement of any particular project or team, nor does it guarantee its security. These reports should not be relied upon in any way by any third party, including for the purpose of making any decision to buy or sell products, services, or any other assets. TO THE FULLEST EXTENT PERMITTED BY LAW, WE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, IN CONNECTION WITH THIS REPORT, ITS CONTENT, RELATED SERVICES AND PRODUCTS, AND YOUR USE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NOT INFRINGEMENT.