# Swingby Skybridge Contract Security Audit

# Table of Contents

## Scope

The scope of the audit is limited to the SwapContract.sol at commit
https://github.com/SwingbyProtocol/skybridge-contract/commit/4efacc185488216f1be78200de542a9145536491

Code of the Skybridge nodes are not included in the scope of this audit.

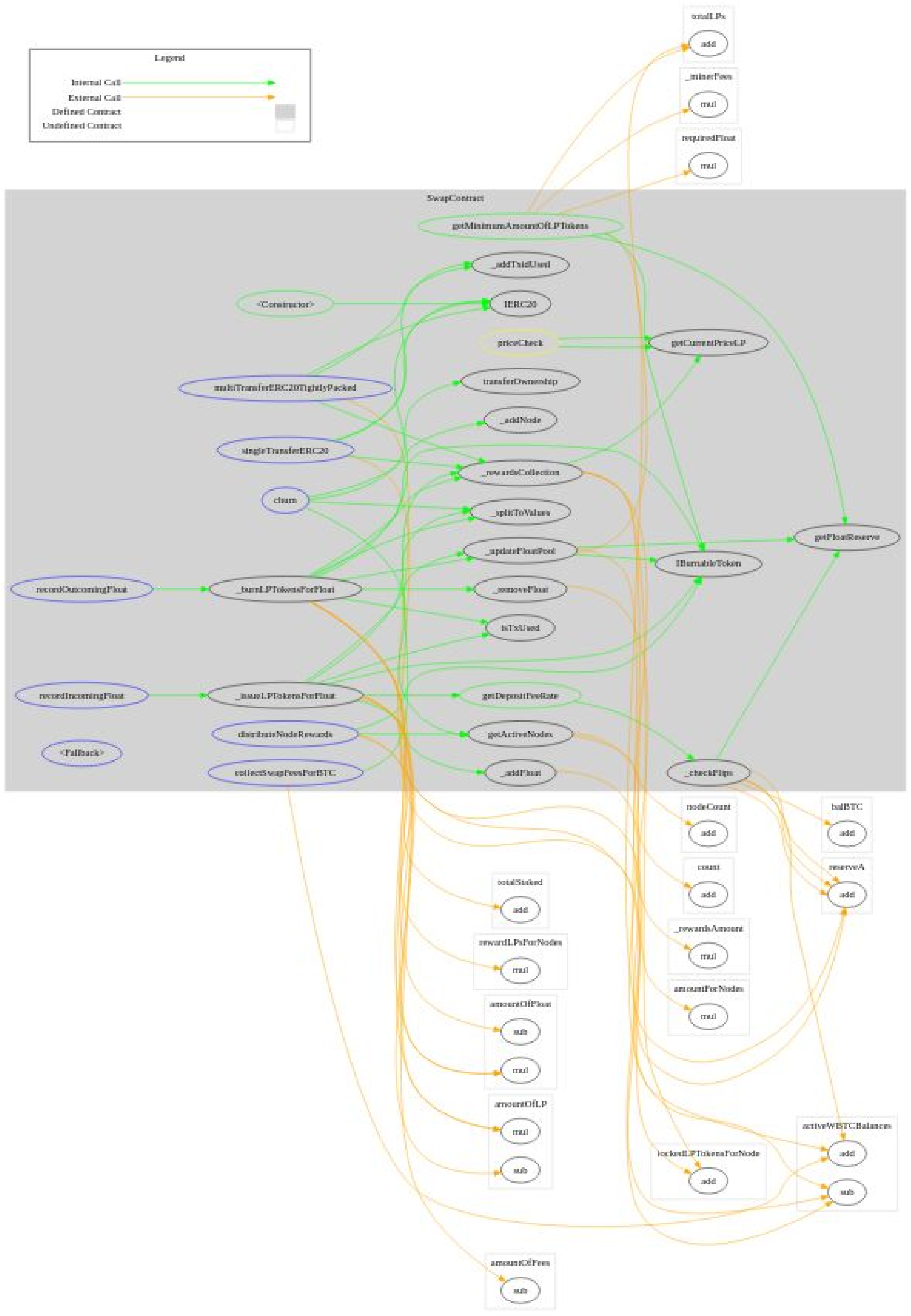
## Summary of Findings

In performing a security audit of the Skybridge SwapContract, several issues of concern were found. For each finding, a summary of the issue is documented, along with any other finer details regarding the issue. Security recommendations are also provided where applicable.

The table below shows a breakdown of security findings found categorized by severity or risk and impact. A finding that has been reported is listed as pending, and if that finding is satisfactorily mitigated, it will be categorized as fixed.

| **Severity** | **Fixed** | **Pending** | **Total** |
|---|---|---|---|
| Critical | 1 | 0 | 1 |
| High | 0 | 0 | 0 |
| Medium | 1 | 0 | 1 |
| Low | 4 | 1 | 5 |
| Info | 7 | 0 | 7 |

## Call graph


Legend
Internal Call
External Call
Defined Contract
Undefined Contract
SwapContract
getMinimumAmountOfLPTokens
_addTxidUsed
<Constructor>
IERC20
priceCheck
getCurrentPriceLP
multiTransferERC20TightlyPacked
transferOwnership
_addNode
singleTransferERC20
_rewardsCollection
churn
_splitToValues
getFloatReserve
_updateFloatPool
IBurnableToken
recordOutcomingFloat
_burnLPTokensForFloat
_removeFloat
isTxUsed
recordIncomingFloat
_issueLPTokensForFloat
getDepositFeeRate
distributeNodeRewards
getActiveNodes
<Fallback>
collectSwapFeesForBTC
_addFloat
_checkFlips
totalLPs
add
_minerFees
mul
requiredFloat
mul
nodeCount
add
balBTC
add
count
add
reserveA
add
totalStaked
add
_rewardsAmount
mul
rewardLPsForNodes
mul
amountForNodes
mul
amountOfFloat
sub
mul
amountOfLP
mul
sub
activeWBTCBalances
add
lockedLPTokensForNode
add
sub
amountOfFees
sub

# Issues

## SBY-001: Externally called only functions can be set as external instead of public

Severity: Info

The following functions are only called externally, never internally, but are currently set as public visibility:

singleTransferERC20(address,address,uint256,uint256,bytes32[]) should be declared external:
    - SwapContract.singleTransferERC20(address,address,uint256,uint256,bytes32[]) (SwapContract.sol#85-96)
multiTransferERC20TightlyPacked(address,bytes32[],uint256,bytes32[]) should be declared external:
    - SwapContract.multiTransferERC20TightlyPacked(address,bytes32[],uint256,bytes32[]) (SwapContract.sol#98-117)
multiTransferERC20(address,address[],uint256[],uint256,bytes32[]) should be declared external:
    - SwapContract.multiTransferERC20(address,address[],uint256[],uint256,bytes32[]) (SwapContract.sol#119-136)
collectSwapFeesForBTC(address,uint256,bytes32) should be declared external:
    - SwapContract.collectSwapFeesForBTC(address,uint256,bytes32) (SwapContract.sol#142-153)
recordIncomingFloat(address,bytes32,bytes32) should be declared external:
    - SwapContract.recordIncomingFloat(address,bytes32,bytes32) (SwapContract.sol#162-174)
recordOutcomingFloat(address,bytes32,bytes32) should be declared external:
    - SwapContract.recordOutcomingFloat(address,bytes32,bytes32) (SwapContract.sol#180-193)
distributeNodeRewards() should be declared external:
    - SwapContract.distributeNodeRewards() (SwapContract.sol#199-218)
churn(address,bytes32[],bool[],uint8,uint8) should be declared external:
    - SwapContract.churn(address,bytes32[],bool[],uint8,uint8) (SwapContract.sol#224-241)
getFloatBalanceOf(address,address) should be declared external:
    - SwapContract.getFloatBalanceOf(address,address) (SwapContract.sol#280-287)
distributeNodeRewards() should be declared external:
    - SwapContract.distributeNodeRewards() (SwapContract.sol#217-236)
deployNewContracts(address,address) should be declared external:
    - SwapContractFactory.deployNewContracts(address,address) (SwapContractFactory.sol#9-19)

Recommendations:

The above mentioned functions can be changed from public to external to reduce gas costs.

Fixed in
https://github.com/SwingbyProtocol/skybridge-contract/commit/773a2139e4ee75c3cbe198e9d584943d6d3f30a5

---

## SBY-002: recordOutcomingFloat lacks onlyOwner modifier

Severity: Critical

The recordOutcomingFloat function is a public function which any account can call to burn an arbitrary amount of LP as well as specify the address for WBTC to be sent to. However, after clarification, it should be a function which can only be called by the TSS node.

Recommendations:
Set the onlyOwner modifier for recordOutcomingFloat.

Fixed in
https://github.com/SwingbyProtocol/skybridge-contract/commit/773a2139e4ee75c3cbe198e9d584943d6d3f30a5

---

## SBY-003: Lack of whitelists for allowed token addresses

Severity: Low

For float related functions, not all have token address validations. As these functions are called by the TSS node, and inputs are expected to be trusted inputs, an additional whitelist check to ensure that only allowed token addresses are passed will help act as an additional layer of security in the smart contract.

Recommendations
Add 0x0 and wBTC into the whitelist. If there are any other additional tokens to be supported in the future, they can be added into the whitelist too.

Fixed in
https://github.com/SwingbyProtocol/skybridge-contract/commit/773a2139e4ee75c3cbe198e9d584943d6d3f30a5

---

## SBY-004: Lack of validation check to ensure that float balance wBTC <= actual wBTC balance in smart contract

Severity: Low

While there is a check that floatAmountOf[token] >= amountOfFloat, in _burnLPTokensForFloat, to ensure that the the amountOfFloat does not exceed the floatAmountOf[token], there is no check in _addFloat and _removeFloat to ensure that the actual wBTC balance in the contract is greater than or equal to the float amount of wBTC tracked.

In the case where there is lesser wBTC balance in the contract, IERC20(token).transfer(to, amountOfFloat) would fail when burning the LP, causing the function to revert.

Recommendations
As there is no normal expected situation where the wBTC balance should be lesser than the tracked float balance of wBTC, it is recommended to add such a check when adding or removing the wBTC balance of the float.

Fixed in
https://github.com/SwingbyProtocol/skybridge-contract/commit/773a2139e4ee75c3cbe198e9d584943d6d3f30a5

---

## SBY-005: Add maximum nodeSize check (100)

Severity: Low

As there are a number of functions that do a for loop using the nodeSize, it would be a good practice to ensure that the nodeSize does not exceed a maximum limit so as to ensure that all the loops can be run without the risk of exceeding the block gas limit.

After discussion with the team, it is expected for the nodeSize to be between 50 to 70, so a limit of 100 would be implementable.

Recommendations
In _addNode, do a check that nodeSize <= 100 before doing an add. This check does not need to be done for remove.

Fixed in
https://github.com/SwingbyProtocol/skybridge-contract/commit/a72648bd6e3d499a02c72f68e52037a927855f8c

---

## SBY-006: Reentrancy prevention

Severity: Info

As the only ERC20 tokens used by the contract are wBTC and the LP token, both of which do not have any callback functionality, reentrancy attacks are not possible. Furthermore, functions that make external calls to the token contracts have the onlyOwner modifier, reentrancy would fail if performed by a callback. Nevertheless, it is recommended as a best practice to make state changes before external calls.

Recommendations
Do the state changes before making the external call.

Fixed in
https://github.com/SwingbyProtocol/skybridge-contract/pull/7/commits/f8ff67fadc586d0cb896cd24aa493f34ae4c2987

---

## SBY-007: Comments as documentation for all functions

Severity: Info

Currently, only the gas usage of each function is commented above each function, but there is a lack of comments describing what each function does, and what each parameter is used for.

Recommendations
Follow a style of comment to document each function to make the code more readable.
E.g. https://docs.soliditylang.org/en/v0.5.8/natspec-format.html

Fixed in
https://github.com/SwingbyProtocol/skybridge-contract/commit/ac1d75942c80ee58f301f60fa48cd672496fdac6

---

## SBY-008: Frontrunning can cause LP deposits to result in having deposit fees deducted

Severity: Medium

The flow of adding a liquidity to the LP is as follows:

1) Making a call to the node API to signal the intent to provide liquidity
2) Making the on-chain transfer

In the case where providing liquidity does not require any fees due to the BTC/wBTC ratio, it is possible to force such a transaction to end up deducting the deposit fee by manipulating the pool ratio before step 2 is made to ensure that the ratio exceeds ⅔.

This does not affect the nodes, but the users who are adding liquidity to the pools.

Recommendations
Add an additional boolean parameter to determine if the fee is expected to be charged.

In the case where there is a change in pool ratio between steps 1 and 2, if the depositFeeZero is set by the TSS node to be true, but the deposit fee is charged, the transaction would be reverted, and left to expire so that the deposit can be refunded.

If the depositFeeZero is set to be false, but the ratio changes such that the deposit fee is not charged, the transaction can be allowed to be processed.

Fixed in
https://github.com/SwingbyProtocol/skybridge-contract/commit/889aa3fd257ae51313c03f889fb252425ec6a9f1

---

## SBY-009: Add check for before ExchangeRate <= after ExchangeRate

Severity: Info

The LP token price increases when swap fees are deducted from swaps that occur. An additional sanity check can be done to ensure that the beforePrice is always lesser than or equal to the afterPrice.

Recommendations:
Add a check comparing the currentExchangeRate before is lesser than or equals to the after for functions that involve fees or LP token minting/burning.

Fixed in
https://github.com/SwingbyProtocol/skybridge-contract/commit/c7660373e00702c02d28fcb35540700636a1bc21

---

## SBY-010: Add checks for parameters of churn function

Severity: Low

churnedInCount and nodeRewardsRatio are both parameters passed when the churn function is called. For the TSS, there are 2 values, t and n, and for a valid signature, there needs to be at least t+1 signers.

Although the TSS logic is maintained beyond the scope of the smart contract, it would be a good idea to check such parameters passed by the TSS node during the call of the churn function.

Recommendations
Add the following checks:
Range checks for 0 <= nodeRewardsRatio <= 100
Add a new parameter for t and ensure that the new value of t >= previous value of t.
churnedInCount (n) has to be checked such that new n >= new t+1

Fixed in
https://github.com/SwingbyProtocol/skybridge-contract/commit/fd32a347870f1febe86ad91bb1a51c2f37abc1c0
https://github.com/SwingbyProtocol/skybridge-contract/commit/974b1d77a41119e086992b1489b8b2751d7fd1e8

---

## SBY-011: Set withdrawalFeeBPS as a state variable

Severity: Info

If withdrawalFeeBPS is only changed when churn is called, it should be set as a public state variable, similar to depositFeesBPS. The removes the need for the TSS node to supply it as a parameter when calling recordOutcomingFloat.

Recommendations:
Set withdrawalFeeBPS as a state variable and set its value in the constructor. If there is a need to allow changing of the value of withdrawalFeeBPS, it can be done in churn or a separate function. Also, if this is done, the check for 0 <= withdrawalFeeBPS <= 100 can be done in the change function.

Fixed in
https://github.com/SwingbyProtocol/skybridge-contract/commit/4efacc185488216f1be78200de542a9145536491
https://github.com/SwingbyProtocol/skybridge-contract/commit/ae700a5d0df0340ab2c678d4126b084f56512615

---

## SBY-012: Add minimum deposit for liquidity deposits

Severity: Low

Even with deposit fees, depending on the float pool ratio, and withdrawal fees, a malicious actor could make small deposits to the float pool. As long as the cost of the transaction to make an wBTC transfer costs lesser than the cost of the transaction by the TSS node to call recordIncomingFloat, it is possible to spend lesser than the TSS node spends, and drain the TSS nodes wallet through numerous small deposits over time.

Recommendations:
The minimum deposit value should be set so as to make it financially expensive for a malicious actor to conduct such an attack.

---

## SBY-013: Prevent overflow for _tssThreshold addition

Severity: Info

When churn is called, _tssThreshold is supplied as a parameter by the TSS node. If the supplied value is 255, which is the max value for uint8, _tssThreshold + uint8(1) would overflow to 0, and  _churnedInCount >= _tssThreshold + uint8(1) would be true even if _tssThreshold is larger than _churnedInCount.

The actual threshold is determined by the tss nodes during key sharing, and not actually used in the smart contract.

Recommendations
Use _tssThreshold.add(uint8(1)) instead to prevent overflow of the uint8.

Fixed in
https://github.com/SwingbyProtocol/skybridge-contract/commit/90ac377ef87169ddb50edc44b8be7391600bee2d
Instead of safemath, a check to ensure that _tssThreshold is lesser than  2**8 - 1.

---

## SBY-014: Add equal array length check for _rewardAddressAndAmounts and _isRemoved

Severity: Info

In the for loop in churn, _rewardAddressAndAmounts.length is used, and it assumes that both _rewardAddressAndAmounts and _isRemoved arrays are of the same length. If the former is shorter than a latter, there would be an out of bounds access of the latter. If the former is longer than the latter, any index that exceeds the former’s length would not be processed for the latter.

Recommendations
Add a check that both arrays are of the same length.

Fixed in
https://github.com/SwingbyProtocol/skybridge-contract/commit/222da2a19161ea5b09ef2f513708959ab895e2f8

---

## Conclusion

Most of the issues raised in the audit have been mitigated, and tests have also been added to the identified cases to ensure that the vulnerable scenarios no longer occur. The scope of this audit is limited only to the Skybridge smart contract, and does not cover the node, so issues such as SBY-012, which are to be resolved in the node were not marked as resolved. Overall, the team was quick to respond to security issues raised and have implemented the security recommendations mentioned in the report.