

# Executive Summary

This audit report was prepared by Quantstamp, the leader in blockchain security.

Type	Yield Protocol	Documentation quality	Low	<div><div></div></div>
Timeline	2025-05-20 through 2025-05-26	Test quality	Low	<div><div></div></div>
Language	Solidity	Total Findings	11	<div><div></div><div>Fixed: 4 Acknowledged: 5 Mitigated: 2</div></div>
Methods	Architecture Review, Unit Testing, Functional Testing, Computer-Aided Verification, Manual Review	High severity findings ⓘ	0	
Specification	Solv Protocol Documentation <a href="#">↗</a>	Medium severity findings ⓘ	2	<div><div></div><div>Fixed: 1 Mitigated: 1</div></div>
Source Code	<ul style="list-style-type: none"><li><a href="https://github.com/solv-finance/SolvBTC">https://github.com/solv-finance/SolvBTC</a> <a href="#">↗</a></li><li><a href="#">#cad7864</a> <a href="#">↗</a></li></ul>	Low severity findings ⓘ	2	<div><div></div><div>Fixed: 1 Acknowledged: 1</div></div>
Auditors	<ul style="list-style-type: none"><li>Adrian Koegl Auditing Engineer</li><li>Leonardo Passos Senior Research Engineer</li><li>Yamen Merhi Auditing Engineer</li></ul>	Undetermined severity findings ⓘ	2	<div><div></div><div>Acknowledged: 2</div></div>
		Informational findings ⓘ	5	<div><div></div><div>Fixed: 2 Acknowledged: 2 Mitigated: 1</div></div>

# Summary of Findings

Quantstamp has completed an audit of new contracts as part of the Solv Protocol. These contracts contains:

- `XSolvBTCOracle` : an oracle to get the latest value of `xSolvBTC` in terms of `SolvBTC` .
- `XSolvBTCPool1` : a pool that allows users to deposit and withdraw `solvBTC` and `xSolvBTC` .
- `SolvBTCRouterV2` : a general purpose router to support automatic routing, automating the multi-step process of depositing and swapping a wrapped bitcoin asset, so the user only needs to perform a single transaction.

Overall, the code is well-written and designed. Technical documentation, however, is largely missing and needs to be urgently improved. Documentation positively correlates with better audit findings and in educating users on potential risks. For this particular audit, the lack of technical documentation was mitigated by active engagement with the Solv team, which promptly assisted in understanding key aspects of the protocol. Part of the tests in the provided suite are currently failing, which raises concerns about the reliability of the testing environment.

No major issues were identified aside from the slippage protection finding ([SOLV-1](#)).

**Fix Review Update:** The client has effectively addressed several findings in their codebase, either by mitigating them like [SOLV-1](#) and [SOLV-7](#), or by fixing them like [SOLV-2](#), [SOLV-3](#), [SOLV-8](#), and [SOLV-9](#). The rest of the findings were acknowledged by the client. However, the test suite still experiences some failing tests.

ID	DESCRIPTION	SEVERITY	STATUS
SOLV-1	Missing Slippage Protection	• Medium ⓘ	Mitigated
SOLV-2	Deposit Expiry Has No Effect	• Medium ⓘ	Fixed
SOLV-3	Unprotected Pools in Edge Case	• Low ⓘ	Fixed
SOLV-4	Unclear Withdraw Process	• Low ⓘ	Acknowledged

ID	DESCRIPTION	SEVERITY	STATUS
SOLV-5	KYC Verification Design May Lead to High Gas Cost	• Informational ⓘ	Acknowledged
SOLV-6	xSolvBTC Price May Be Outdated	• Informational ⓘ	Acknowledged
SOLV-7	Single Admin Role for Many Responsibilities	• Informational ⓘ	Mitigated
SOLV-8	admin Can Arbitrarily Set the Price of xSolvBTC	• Informational ⓘ	Fixed
SOLV-9	Inconsistent Storage Layout Patterns Across Contracts	• Informational ⓘ	Fixed
SOLV-10	Pool Restrictions Can Be Potentially Bypassed	• Undetermined ⓘ	Acknowledged
SOLV-11	Pool Permissions Not Enforced on Withdrawals	• Undetermined ⓘ	Acknowledged

# Assessment Breakdown

Quantstamp's objective was to evaluate the repository for security-related issues, code quality, and adherence to specification and best practices.

i

Disclaimer

Only features that are contained within the repositories at the commit hashes specified on the front page of the report are within the scope of the audit and fix review. All features added in future revisions of the code are excluded from consideration in this report.

Possible issues we looked for included (but are not limited to):

- Transaction-ordering dependence
- Timestamp dependence
- Mishandled exceptions and call stack limits
- Unsafe external calls
- Integer overflow / underflow
- Number rounding errors
- Reentrancy and cross-function vulnerabilities
- Denial of service / logical oversights
- Access control
- Centralization of power
- Business logic contradicting the specification
- Code clones, functionality duplication
- Gas usage
- Arbitrary token minting

Methodology

1. Code review that includes the following
  1. Review of the specifications, sources, and instructions provided to Quantstamp to make sure we understand the size, scope, and functionality of the smart contract.
  2. Manual review of code, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
  3. Comparison to specification, which is the process of checking whether the code does what the specifications, sources, and instructions provided to Quantstamp describe.
2. Testing and automated analysis that includes the following:
  1. Test coverage analysis, which is the process of determining whether the test cases are actually covering the code and how much code is exercised when we run those test cases.
  2. Symbolic execution, which is analyzing a program to determine what inputs cause each part of a program to execute.
3. Best practices review, which is a review of the smart contracts to improve efficiency, effectiveness, clarity, maintainability, security, and control based on the established industry and academic practices, recommendations, and research.
4. Specific, itemized, and actionable recommendations to help you take steps to secure your smart contracts.

# Scope

Files Included

Repo: <https://github.com/solv-finance/SolvBTC>

# Operational Considerations

- We assume all dependencies to be secure and behave as intended. Particularly, `OpenFundMarket` and `MultiAssetPool` .
- We assume that the `admin` correctly configures all dependencies to non-malicious contracts.
- We assume the `admin` to regularly and honestly post the price of `xSolvBTC` .
- We assume that the private keys of privileged actors are assumed to be securely managed in accordance to industry best practices.
- This audit does not guarantee the behavior of future contract upgrades of these contracts. We assume all future upgrades will go through thorough audits.

# Key Actors And Their Capabilities

## SolvBTCRouterV2:

- `owner` :
  - Sets the `OpenFundMarket` address
  - Configures token-to-pool mappings
  - Defines conversion paths between tokens
  - Links tokens to their respective `MultiAssetPools`
  - Manages KYC SBT verifiers

## XSolvBTCOracle:

- `admin` :
  - Sets the NAV (Net Asset Value) of `xSolvBTC`
  - Configures the `xSolvBTC` token address
  - Controls oracle precision (decimals)

## XSolvBTCPool:

- `admin` :
  - Sets the fee recipient address
  - Configures withdrawal fee rates
  - Controls deposit allowance (can pause deposits)

# Findings

## SOLV-1 Missing Slippage Protection

• Medium ⓘ Mitigated

✓ Update

Marked as "Mitigated" by the client.  
Addressed in: `6bc9b556db6ef03f89c1a9b4cfcf5eb890a5b315` , `0df241acd76f3e60e539857cb3c4c52b1b43b08f` , `ba1093f18f4695135e18bb84096782d96c3b5bc0` .  
The client provided the following explanation:  
  

We have added some restrictions to the deposit/withdraw functions in the `XSolvBTCPool` contract to limit the receivable token amount. In addition, `xSolvBTC` is designed with a monotonically increasing nav, guaranteeing that the amount of tokens users redeem will never be less than their initial deposit. Thus, the slippage will not result in user losses.  
The deposit function in the `SolvBTCRouterV2` contract has been updated by adding the `minimumTargetTokenAmount_` parameter to enforce a lower bound on the amount of the target token that users will receive.  
We will keep the deposit and withdraw functions in the `XSolvBTCPool` contract unchanged for two reasons: First, all user-facing deposits are processed through the `SolvBTCRouterV2` contract, which may involve multiple token conversions and therefore requires final validation in the `SolvBTCRouterV2` contract. Second, we prioritize successful user withdrawals over minimizing slippage impact.

File(s) affected: `contracts/XSolvBTCPool.sol` , `contracts/SolvBTCRouterV2.sol`

**Description:** Due to swap price fluctuations, depositing through the router may lead to a less than anticipated amount. Similarly, users depositing or withdrawing from `XSolvBTCPool` may be affected by unfavourable conversion rates and withdrawal fee increases.

**Recommendation:** Allow users to specify a minimum acceptable return amount for deposits and withdrawals, reverting the transaction if slippage exceeds expectations.

## SOLV-2 Deposit Expiry Has No Effect

• Medium ⓘ Fixed

✓ **Update**

Marked as "Fixed" by the client.  
Addressed in: `96b8ffb1a2fa868efb4b345304099f4b200c4d90` .

**File(s) affected:** `contracts/SolvBTCRouterV2.sol`

**Description:** The router's deposit function limits the transaction expiry to a maximum of 5 minutes, i.e., `block.timestamp + 300` . The actual stale check occurs in `IOpenFundMarket.subscribe()` , where the `expireTime_` is compared to `block.timestamp` . Since both the router and the market functions execute within the same transaction block, this check always passes and fails to enforce a mempool lifetime limit.

**Recommendation:** Allow users to specify the expiry time as a deposit parameter. Ensure the latter is validated.

## SOLV-3 Unprotected Pools in Edge Case

• Low ⓘ Fixed

✓ **Update**

Marked as "Fixed" by the client.  
Addressed in: `331a5eed6d2444f7c1f543b6d004ce67bbcc205f` .

**File(s) affected:** `contracts/SolvBTCRouterV2.sol`

**Description:** Some pools are permissioned and protected through a whitelisting and KYC mechanism. If the address is whitelisted OR is in possession of a registered KYC soul-bound token, the user may use the selected pool. However, if the array of `kycSBTVerifiers` is empty, meaning no SBT were registered yet or all were removed, anyone can effectively access all pools.

**Recommendation:** Consider whether this is intended. If not, return `false` in `checkKycSBT()` at the end of the function.

## SOLV-4 Unclear Withdraw Process

• Low ⓘ Acknowledged

ⓘ **Update**

Marked as "Acknowledged" by the client.  
The client provided the following explanation:

No modification is necessary. In `xSolvBTC` case, users should interact with the `withdraw()` function in the `xSolvBTCPool` contract to get back the underlying asset `SolvBTC`. In other cases, users can interact with the `withdrawRequest()` function in the `SolvBTCRouterV2` contract and receive an redemption `sft` token, from which they can claim underlying assets later.

**File(s) affected:** `contracts/SolvBTCRouterV2.sol`

**Description:** The process for users to withdraw their deposited assets is unclear. While users receive a redemption token upon withdrawal request, the mechanism for converting that token back into the underlying deposited asset is not evident.

**Recommendation:** Provide clear user-facing documentation that outlines the complete withdrawal process and any associated risks.

## SOLV-5 KYC Verification Design May Lead to High Gas Cost

• Informational ⓘ Acknowledged

ⓘ **Update**

Marked as "Acknowledged" by the client.  
The client provided the following explanation:

No modification is necessary. The number of registered KYC SBTs will be strictly capped, so this verification process has minimal impact on gas consumption.


**File(s) affected:** `contracts/SolvBTCRouterV2.sol`

**Description:** The current design of the KYC process requires the `checkKycSBT()` to loop through all verifiers, in the worst case. With an increasing amount of users and an increasing number of registered SBTs, this will utilize an increasing amount of gas.

**Recommendation:** We recommend changing the design such that verified SBTs can be verified through a mapping. Particularly, we suggest maintaining a mapping of user addresses to an array of SBTs. When a new SBT is registered, it can be added to the array of the address owning it. This design works since SBTs cannot change the owner.

SOLV-6xSolvBTCPrice May Be Outdated

• Informational ⓘAcknowledged



**Update**  
Marked as "Acknowledged" by the client.  
The client provided the following explanation:  

No modification is necessary. The xSolvBTCOracle contract inherits from the ISolvBTCYieldTokenOracle contract, which has already been deployed, so the return value of the getNav() function cannot be changed. Additionally, the xSolvBTCOracle contract implements the latestUpdatedAt() function to verify whether the returned price value is outdated.


**File(s) affected:** contracts/oracle/XSolvBTCOracle.sol

**Description:** The XSolvBTCOracle should keep track of the price of xSolvBTC in terms of SolvBTC . According to in-line documentation, the price will be updated by the admin approximately every week. However, if the price is outdated because the update is (much) older than a week, protocols utilizing xSolvBTC will not be prevented from using the old conversion rate.

**Recommendation:** A common practice for oracles is to provide the last updated timestamp as part of the getNav() call or even return a boolean on whether the price is outdated. Furthermore, we recommend explicitly defining and documenting the expiry of an oracle date.

SOLV-7Single Admin Role for Many Responsibilities

• Informational ⓘMitigated



**Update**  
Marked as "Fixed" by the client.  
Addressed in: 4490c7b7c68f74a3e921ead54e0e4be89eba4a37 .  
The client provided the following explanation:  

The modified SolvBTCRouterV2 contract has eliminated the setOpenFundMarket() function, and the market contract address configuration has been moved to the initialize() function to enhance security. This simplifies the permission management in SolvBTCRouterV2 contract, requiring a single admin role to perform daily operations including setPath(), setPoolId() and setMultiAssetPool().


**File(s) affected:** contracts/SolvBTCRouterV2.sol

**Description:** The SolvBTCRouterV2 uses one admin role through Ownable to execute admin functionalities. However, some of the functions, such as setPath() or setPoolId() may be used more frequently than, e.g, setMultiAssetPool() , setOpenFundMarket() , and, specifically, transferOwnership() . Due to the regular utilization of the keys, the risk of key compromise is increased. Since this admin can also transfer the ownership, the loss of keys could lead to the ultimate loss of control over the contract as well.

**Recommendation:** It is best practices to separate admin roles such that a "hot key" exists that performs daily operations while a "cold key" can recover from key loss or perform more sensitive and rare operations such as setOpenFundMarket() .

SOLV-8adminCan Arbitrarily Set the Price ofxSolvBTC

• Informational ⓘFixed



**Update**  
Marked as "Fixed" by the client.  
Addressed in: b7e440873c75945e6068ddeb97853cf3224956ec .  
The client provided the following explanation:  

Modified as recommended. We have added some restrictions to the setNav() function in the XSolvBTCOracle contract to ensure that the nav can only increase and the nav growth should be within a specific range.



**i Update**

The initial recommendation was to allow only monotonic increases in price. However, after further review, we revised the recommendation to support limited downward adjustment within a defined deviation range. This adjustment accounts for scenarios where the price may have been incorrectly increased due to human or script errors. Without the ability to decrease the price in such cases, the asset could remain artificially overvalued, leading to inaccurate valuations for deposits and withdrawals. The client is aware of this limitation but chose to maintain a strictly monotonic-increasing pricing model.

**File(s) affected:** `contracts/oracle/XSolvBTCOracle.sol`

**Description:** The price of `xSolvBTC` in terms of `SolvBTC` is entirely determined by the `admin` of `XSolvBTCOracle`. Therefore, additional user trust is required to stake their `SolvBTC`.

**Recommendation:** In normal operating conditions, `xSolvBTC`'s price should rise monotonically relative to `SolvBTC`. An edge case arises when a prior update has inadvertently set the NAV above its true value; in such cases, a downward correction is required.

We therefore recommend allowing `setNav()` to accept new values only within a narrow, configurable deviation band (e.g.,  $\pm 8\%$  of the previous NAV), enabling controlled decreases to fix these rare occasional over-valuations while preventing large, arbitrary price movements. If implementing this deviation band is impractical, enforcing a strictly monotonic-increasing rule remains an acceptable fallback.

**SOLV-9**  
**Inconsistent Storage Layout Patterns Across Contracts**

• **Informational** ⓘ **Fixed**

**✓ Update**

Marked as "Fixed" by the client.  
Addressed in: `61715c261bc2b82392ede3e42d84bf6d18e541f0`.  
The client provided the following explanation:

The modified `XSolvBTCOracle` contract has implemented the `__gap` pattern for future upgrades. However, since the previous versions of these contracts have already been deployed to several chains, the current upgradeable patterns should to be retained.

**File(s) affected:** `contracts/SolvBTCRouterV2.sol`, `contracts/oracle/XSolvBTCOracle.sol`, `contracts/XSolvBTCPool.sol`

**Description:** The codebase uses inconsistent usage of storage layout patterns across upgradeable contracts:

- `XSolvBTCPool` uses the EIP-7201 namespaced storage layout, which is the recommended modern approach for upgradeable contracts.
- `SolvBTCRouterV2` relies on the traditional `__gap` pattern to reserve storage slots for future upgrades.
- `XSolvBTCOracle` inherits from upgradeable modules but does not implement either the `__gap` pattern or namespaced storage layout, which increases the risk of storage collisions during future upgrades.

**Recommendation:** Unify the storage layout approach across all upgradeable contracts. Prefer adopting EIP-7201 namespaced storage layout for consistency, improved readability, and safer upgrade paths.

**SOLV-10 Pool Restrictions Can Be Potentially Bypassed**

• **Undetermined** ⓘ **Acknowledged**

**i Update**

Marked as "Acknowledged" by the client.  
The client provided the following explanation:

No modification is necessary. There are two kinds pool restrictions in the `SolvBTCRouterV2` contract. The first one is permissionless and whitelisted check, which derives from the `OpenFundMarket` contract, so they share the same access restrictions. The other one is KYC verification, for those who are KYC-approved but not whitelisted, the `SolvBTCRouterV2` contract is the only way they can deposit through, no way can they bypass the KYC verification by interacting directly with the `OpenFundMarket` contract.

**File(s) affected:** `contracts/SolvBTCRouterV2.sol`

**Description:** Deposits through the router requires pools to be either permissionless, whitelisted, or KYC-approved. However, users may bypass these checks by interacting directly with the pool contract, assuming it doesn't replicate the same restrictions as the router.

**Recommendation:** Ensure pools independently enforce the same access restrictions as the router.

**SOLV-11 Pool Permissions Not Enforced on Withdrawals**

• **Undetermined** ⓘ **Acknowledged**

**i Update**

Marked as "Acknowledged" by the client.  
The client provided the following explanation:

No modification is necessary. Withdrawals do not need pool permission checks.

**File(s) affected:** `contracts/SolvBTCRouterV2.sol`

**Description:** The same pool permission checks enforced during deposits should be applied during withdrawals. Currently, such a check is missing.

**Recommendation:** Add permission checks in the `withdrawRequest()` function to match those in the deposit flow.

# Auditor Suggestions

## S1 Gas Optimization

Fixed

**✓ Update**

Marked as "Fixed" by the client.  
Addressed in: `13cc8940b3b4b24853a782705dbae2ad5c4dbfa1` .  
The client provided the following explanation:

The openzeppelin version applied in this project is v5.0.1, where `ReentrancyGuardTransientUpgradeable` contract is not included.

**File(s) affected:** `contracts/SolvBTCRouterV2.sol` , `contracts/XSolvBTCPool.sol` , `contracts/oracle/XSolvBTCOracle.sol`

**Description:** Several parts of the codebase can benefit from minor gas optimizations:

1. Replace `ReentrancyGuardUpgradeable` with `ReentrancyGuardTransientUpgradeable` to reduce gas overhead for transient reentrancy checks.
2. In `Oracle.setNav()` , consider emitting `block.timestamp` directly instead of using `_latestUpdatedAt` variable.
3. In `Oracle.getNav()` , cache the `_latestNav` storage variable into memory.
4. In `XSolvBTCPool.deposit()` and `withdraw()` , cache `$.xSolvBTC` and `$.solvBTC` in local memory variables.
5. In `SolvBTCRouterV2._deposit()` , `withdraw()` , `cancelWithdrawRequest()` , and `checkPoolPermission()` , cache the `openFundMarket` storage variable in memory.

**Recommendation:** Implement the above changes to reduce gas consumption.

## S2 Code Improvements

Fixed

**✓ Update**

Marked as "Fixed" by the client.  
Addressed in: `45124368abbe4870cac3bac605617a1423678d1b` .

**File(s) affected:** `contracts/SolvBTCRouterV2.sol` , `contracts/oracle/XSolvBTCOracle.sol` , `contracts/XSolvBTCPool.sol`

**Description:** The codebase can benefit from several improvements:

1. The `initialize()` function of `XSolvBTCPool` redundantly checks if `feeRecipient_` is not `address(0)` . This check is already performed in the `_setFeeRecipient()` function.
2. The dev comments in the head of the `XSolvBTCPool` can be improved to describe the correct process of burning and minting `SolvBTC` and `xSolvBTC` :
  - Deposit: Burns SolvBTC, mints xSolvBTC.
  - Withdraw: Burns xSolvBTC, mints SolvBTC.
3. Hardcoded values reduce code clarity and should be replaced with named constant variables, properly documenting their purpose:
  - 300 seconds expiry in `SolvBTCRouterV2._deposit()`
  - `86400 * 86400` in `XSolvBTCOracle._getDate()`
4. The codebase lacks adequate inline comments and public documentation. This impairs maintainability and auditing. All invariants, function purposes, and data flow assumptions should be documented.
5. In the `XSolvBTCPool` , most error messages contain the string "SolvBTCMultiAssetPool: ..." , but should start with "XSolvBTCPool: ...".

**Recommendation:** Consider implementing the suggested changes.

# Definitions

- **High severity** – High-severity issues usually put a large number of users' sensitive information at risk, or are reasonably likely to lead to catastrophic impact for client's reputation or serious financial implications for client and users.
- **Medium severity** – Medium-severity issues tend to put a subset of users' sensitive information at risk, would be detrimental for the client's reputation if exploited, or are reasonably likely to lead to moderate financial impact.
- **Low severity** – The risk is relatively small and could not be exploited on a recurring basis, or is a risk that the client has indicated is low impact in view of the client's business circumstances.
- **Informational** – The issue does not pose an immediate risk, but is relevant to security best practices or Defence in Depth.
- **Undetermined** – The impact of the issue is uncertain.
- **Fixed** – Adjusted program implementation, requirements or constraints to eliminate the risk.
- **Mitigated** – Implemented actions to minimize the impact or likelihood of the risk.
- **Acknowledged** – The issue remains in the code but is a result of an intentional business or design decision. As such, it is supposed to be addressed outside the programmatic means, such as: 1) comments, documentation, README, FAQ; 2) business processes; 3) analyses showing that the issue shall have no negative consequences in practice (e.g., gas analysis, deployment settings).

# Appendix

### File Signatures

The following are the SHA-256 hashes of the reviewed files. A file with a different SHA-256 hash has been modified, intentionally or otherwise, after the security review. You are cautioned that a different SHA-256 hash could be (but is not necessarily) an indication of a changed condition or potential vulnerability that was not within the scope of the review.

### Files

- Repo: `https://github.com/solv-finance/SolvBTC`
- `5ff...ca7 ./contracts/SolvBTCRouterV2.sol`
- `718...deb ./contracts/XSolvBTCPool.sol`
- `e33...bee ./contracts/oracle/XSolvBTCOracle.sol`

# Test Suite Results

The test suite experienced some failing tests. 115 tests executed successfully and 37 failing tests.

**Fix Review Update:** The test suite still experiences failures: 115 tests pass, while 45 tests fail.

```
Ran 1 test for test/SolvBTCYieldTokenV2_1.t.sol:SolvBTCYieldTokenV2Test
[FAIL: setup failed: EvmError: Revert] setUp() (gas: 0)
Suite result: FAILED. 0 passed; 1 failed; 0 skipped; finished in 4.66ms (0.00ns CPU time)

Ran 1 test for test/SolvBTCYieldTokenV3.t.sol:SolvBTCYieldTokenV3Test
[FAIL: setup failed: EvmError: Revert] setUp() (gas: 0)
Suite result: FAILED. 0 passed; 1 failed; 0 skipped; finished in 4.82ms (0.00ns CPU time)

Ran 1 test for test/SolvBTCYieldTokenV3_1.t.sol:SolvBTCYieldTokenV3_1Test
[FAIL: setup failed: EvmError: Revert] setUp() (gas: 0)
Suite result: FAILED. 0 passed; 1 failed; 0 skipped; finished in 4.67ms (0.00ns CPU time)

Ran 1 test for test/SolvBTCV3.t.sol:SolvBTCV3Test
[FAIL: setup failed: EvmError: Revert] setUp() (gas: 0)
Suite result: FAILED. 0 passed; 1 failed; 0 skipped; finished in 5.33ms (0.00ns CPU time)

Ran 1 test for test/SolvBTCV3_1.t.sol:SolvBTCV3_1Test
[FAIL: setup failed: EvmError: Revert] setUp() (gas: 0)
Suite result: FAILED. 0 passed; 1 failed; 0 skipped; finished in 5.26ms (0.00ns CPU time)

Ran 1 test for test/SolvBTCV2_1.t.sol:SolvBTCV2_1Test
[FAIL: setup failed: EvmError: Revert] setUp() (gas: 0)
Suite result: FAILED. 0 passed; 1 failed; 0 skipped; finished in 156.13ms (0.00ns CPU time)
```



```
Ran 4 tests for test/SolvBTCRouterV2.t.sol:SolvBTCRouterV2Test
[FAIL: EvmError: Revert] test_CancelWithdrawRequest() (gas: 5653)
[FAIL: EvmError: Revert] test_Deposit_WBTC_for_SOLVBTC() (gas: 5651)
[FAIL: EvmError: Revert] test_Deposit_WBTC_for_SOLVBTCBBN() (gas: 5607)
[FAIL: EvmError: Revert] test_WithdrawRequest() (gas: 5652)
Suite result: FAILED. 0 passed; 4 failed; 0 skipped; finished in 156.38ms (253.00µs CPU time)
```

```
Ran 2 tests for test/NativeDeposit.t.sol:NativeDepositTest
[FAIL: EvmError: Revert] test_NativeDeposit() (gas: 7751)
[PASS] test_NativeDepositStatus() (gas: 19500)
Suite result: FAILED. 1 passed; 1 failed; 0 skipped; finished in 156.41ms (68.29µs CPU time)
```

```
Ran 9 tests for test/SolvBTCFactoryV3.t.sol:SolvBTCFactoryV3Test
[PASS] test_DeployProduct() (gas: 595315)
[PASS] test_RevertWhenDeployBeaconByNonAdmin() (gas: 17484)
[PASS] test_RevertWhenDeployProductByNonGovernor() (gas: 287607)
[PASS] test_RevertWhenRemoveProductByNonAdmin() (gas: 582127)
[PASS] test_RevertWhenTransferBeaconOwnershipByNonAdmin() (gas: 285270)
[PASS] test_RevertWhenUpgradeBeaconByNonAdmin() (gas: 1913568)
[PASS] test_Role() (gas: 15001)
[PASS] test_TransferBeaconOwnership() (gas: 295775)
[PASS] test_UpgradeBeacon() (gas: 2728484)
Suite result: ok. 9 passed; 0 failed; 0 skipped; finished in 157.54ms (1.60ms CPU time)
```

```
Ran 11 tests for test/SolvBTCFactory.t.sol:SolvBTCFactoryTest
[PASS] test_DeployProduct() (gas: 544057)
[PASS] test_ImportBeaconFromPreviousSWTFactory() (gas: 1957651)
[PASS] test_ImportProxyFromPreviousSWTFactory() (gas: 1747654)
[PASS] test_RevertWhenDeployBeaconByNonAdmin() (gas: 17506)
[PASS] test_RevertWhenDeployProductByNonGovernor() (gas: 287298)
[PASS] test_RevertWhenRemoveProductByNonAdmin() (gas: 530943)
[PASS] test_RevertWhenTransferBeaconOwnershipByNonAdmin() (gas: 285323)
[PASS] test_RevertWhenUpgradeBeaconByNonAdmin() (gas: 1878542)
[PASS] test_Role() (gas: 15023)
[PASS] test_TransferBeaconOwnership() (gas: 295821)
[PASS] test_UpgradeBeacon() (gas: 2677332)
Suite result: ok. 11 passed; 0 failed; 0 skipped; finished in 153.65ms (2.39ms CPU time)
```

```
Ran 10 tests for test/SftWrappedTokenFactory.t.sol:SftWrappedTokenFactoryTest
[PASS] test_DeployProduct() (gas: 766324)
[PASS] test_RevertWhenDeployBeaconByNonAdmin() (gas: 46739)
[PASS] test_RevertWhenDeployProductByNonGovernor() (gas: 290217)
[PASS] test_RevertWhenRemoveProductByNonGovernor() (gas: 739824)
[PASS] test_RevertWhenSetImplementationByNonAdmin() (gas: 16695)
[PASS] test_RevertWhenTransferBeaconOwnershipByNonAdmin() (gas: 287700)
[PASS] test_RevertWhenUpgradeBeaconByNonAdmin() (gas: 2417871)
[PASS] test_RoleSet() (gas: 15048)
[PASS] test_TransferBeaconOwnership() (gas: 298474)
[PASS] test_UpgradeBeacon() (gas: 2889459)
Suite result: ok. 10 passed; 0 failed; 0 skipped; finished in 170.82ms (1.61ms CPU time)
```

```
Ran 27 tests for test/SolvBTCMultiAssetPool.t.sol:SolvBTCMultiAssetPoolTest
[PASS] test_AddSftSlot() (gas: 238065)
[FAIL: panic: division or modulo by zero (0x12)] test_FullDepositWhenHoldingValueSftIdIsNotZero() (gas: 442992)
[FAIL: revert: ERC3525: invalid token ID] test_FullDepositWhenHoldingValueSftIdIsZero() (gas: 212496)
[PASS] test_InitialStatus() (gas: 33519)
[PASS] test_PartialDepositWhenHoldingValueSftIdIsNotZero() (gas: 532512)
[FAIL: revert: ERC3525: invalid token ID] test_PartialDepositWhenHoldingValueSftIdIsZero() (gas: 212576)
[PASS] test_RemoveSftSlot() (gas: 205382)
[PASS] test_RevertWhenAddExistedSftSlot() (gas: 192418)
[PASS] test_RevertWhenAddSftSlotByNonAdmin() (gas: 18953)
[PASS] test_RevertWhenAddSftSlotWithSftIdNotOwned() (gas: 62076)
[PASS] test_RevertWhenAddSftSlotWithSftIdOfMismatchedSlot() (gas: 56339)
[PASS] test_RevertWhenDecimalsNotMatched() (gas: 47415)
[PASS] test_RevertWhenDepositToInvalidSftSlot() (gas: 85560)
[PASS] test_RevertWhenDepositValueExceedsSftIdBalance() (gas: 251135)
[PASS] test_RevertWhenDepositValueIsZero() (gas: 232010)
[PASS] test_RevertWhenDepositWithSftIdNotOwned() (gas: 242410)
[PASS] test_RevertWhenWithdrawButPoolHoldingValueNotEnough() (gas: 236931)
[FAIL: Error != expected error: SolvBTCMultiAssetPool: withdraw amount cannot be 0 != SolvBTCMultiAssetPool: insufficient balance] test_RevertWhenWithdrawButPoolHoldingValueSftIdIsZero()
```

```
(gas: 207242)
[FAIL: Error != expected error: SolvBTCMultiAssetPool: withdraw amount cannot be 0 !=
SolvBTCMultiAssetPool: sft slot not allowed] test_RevertWhenWithdrawToInvalidSftSlot() (gas: 41582)
[FAIL: Error != expected error: SolvBTCMultiAssetPool: withdraw amount cannot be 0 !=
SolvBTCMultiAssetPool: caller is not sft owner] test_RevertWhenWithdrawToSftIdNotOwned() (gas: 207309)
[PASS] test_RevertWhenWithdrawValueExceedsBalance() (gas: 246011)
[PASS] test_RevertWhenWithdrawValueIsZero() (gas: 208578)
[FAIL: Error != expected error: ERC3525: invalid token ID != SolvBTCMultiAssetPool: slot not matched]
test_RevertWhenWithdrawWithMismatchedSpecifiedSlot() (gas: 498284)
[PASS] test_RevertWhenchangeSftSlotAllowedByNonAdmin() (gas: 191970)
[FAIL: revert: SolvBTCMultiAssetPool: withdraw amount cannot be 0]
test_WithdrawForThoseWhoDepositBeforeUpgrade() (gas: 216716)
[FAIL: panic: division or modulo by zero (0x12)] test_WithdrawSlotAAfterDepositSlotA() (gas: 427065)
[FAIL: revert: ERC3525: invalid token ID] test_WithdrawSlotBAfterDepositSlotA() (gas: 453444)
Suite result: FAILED. 17 passed; 10 failed; 0 skipped; finished in 165.25ms (16.56ms CPU time)
```

```
Ran 25 tests for test/SolvBTC.t.sol:SolvBTCTest
[PASS] test_AccessControlInitialStatus() (gas: 3951262)
[PASS] test_BurnBySolvBTCMultiAssetPool() (gas: 3955450)
[PASS] test_BurnWithAccountByAnotherMinter() (gas: 3985124)
[PASS] test_BurnWithoutAccountByAnotherMinter() (gas: 3984850)
[PASS] test_ERC165() (gas: 3937498)
[PASS] test_GrantAdminRole() (gas: 3968840)
[PASS] test_GrantMinterRole() (gas: 3969983)
[PASS] test_MintByAnotherMinter() (gas: 3985141)
[PASS] test_MintBySolvBTCMultiAssetPool() (gas: 3955337)
[PASS] test_OwnershipInitialStatus() (gas: 3937134)
[PASS] test_RenounceAdminRole() (gas: 3955453)
[PASS] test_RevertWhenAcceptOwnershipByNonPendingAdmin() (gas: 3956679)
[PASS] test_RevertWhenBurnWithAccountByNonPoolBurner() (gas: 3942423)
[PASS] test_RevertWhenBurnWithoutAccountByNonMinter() (gas: 3942142)
[PASS] test_RevertWhenCallInitializeRepeatedly() (gas: 3932664)
[PASS] test_RevertWhenCallInitializeV2Repeatedly() (gas: 3932057)
[PASS] test_RevertWhenGrantRoleByNonAdminRole() (gas: 3969149)
[PASS] test_RevertWhenMintByNonMinter() (gas: 3942477)
[PASS] test_RevertWhenReceivingERC3525() (gas: 4221374)
[PASS] test_RevertWhenReceivingERC721() (gas: 4015010)
[PASS] test_RevertWhenTransferOwnershipByNonAdmin() (gas: 3932882)
[PASS] test_RevokeAdminRole() (gas: 3956043)
[PASS] test_SolvBTCStatusAfterUpgrade() (gas: 3996303)
[FAIL: panic: division or modulo by zero (0x12)] test_SweepEmptySftIds() (gas: 3998437)
[PASS] test_TransferOwnership() (gas: 3947021)
Suite result: FAILED. 24 passed; 1 failed; 0 skipped; finished in 170.64ms (344.95ms CPU time)
```

```
Ran 14 tests for test/SftWrapRouter.t.sol:SftWrapRouterTest
[PASS] test_CancelRedemption() (gas: 2130952)
[PASS] test_CreateRedemption() (gas: 1493990)
[PASS] test_CreateSubscription() (gas: 697648)
[PASS] test_FirstStakeWithAllValue() (gas: 365262)
[PASS] test_FirstStakeWithPartialValue() (gas: 853525)
[PASS] test_InitialStatus() (gas: 25322)
[PASS] test_NonFirstStakeWithAllValue() (gas: 676216)
[PASS] test_NonFirstStakeWithPartialValue() (gas: 951511)
[PASS] test_OnERC3525Received_FirstStake() (gas: 765108)
[PASS] test_OnERC3525Received_NotFirstStake() (gas: 864513)
[PASS] test_OnERC721Received_FirstStake() (gas: 337041)
[PASS] test_OnERC721Received_NotFirstStake() (gas: 593337)
[PASS] test_UnstakeWhenGivenSftId() (gas: 739022)
[PASS] test_UnstakeWhenNotGivenSftId() (gas: 932826)
Suite result: ok. 14 passed; 0 failed; 0 skipped; finished in 170.80ms (43.74ms CPU time)
```

```
Ran 19 tests for test/SftWrappedToken.t.sol:SftWrappedTokenTest
[PASS] test_BurnWithGivenSftId() (gas: 329937)
[PASS] test_BurnWithoutGivenSftIdWhenHoldingEmptySftIdsIsBlank() (gas: 572833)
[PASS] test_BurnWithoutGivenSftIdWhenHoldingEmptySftIdsIsNotBlank() (gas: 744759)
[PASS] test_InitialMintWithAllValue() (gas: 272799)
[PASS] test_InitialMintWithPartialValue() (gas: 501767)
[PASS] test_InitialTransferWithId() (gas: 254176)
[PASS] test_InitialTransferWithValue() (gas: 471022)
[PASS] test_NonInitialMintWithAllValue() (gas: 467861)
[PASS] test_NonInitialMintWithPartialValue() (gas: 345756)
[PASS] test_NonInitialTransferWithId() (gas: 449146)
```

```
[PASS] test_NonInitialTransferWithValue() (gas: 316716)
[PASS] test_RevertWhenBurnWithSftIdNotOwned() (gas: 259581)
[PASS] test_RevertWhenBurnWithSftIdOfInvalidSlot() (gas: 261837)
[PASS] test_RevertWhenBurnWithZeroAmount() (gas: 245218)
[PASS] test_RevertWhenDirectlyCallOnERC3525ReceivedFunction() (gas: 46938)
[PASS] test_RevertWhenDirectlyCallOnERC721ReceivedFunction() (gas: 23242)
[PASS] test_RevertWhenTransferIdOfInvalidSlot() (gas: 169782)
[PASS] test_RevertWhenTransferValueOfInvalidSlot() (gas: 383394)
[PASS] test_RevertWhenTransferValueToNonHoldingValueSftId() (gas: 964731)
Suite result: ok. 19 passed; 0 failed; 0 skipped; finished in 171.36ms (24.55ms CPU time)
```

Ran 23 tests for test/SolvBTCRouter.t.sol:SolvBTCRouterTest

```
[FAIL: panic: division or modulo by zero (0x12)] test_CancelRedemption() (gas: 835383)
[FAIL: panic: division or modulo by zero (0x12)] test_CreateRedemption() (gas: 835424)
[FAIL: panic: division or modulo by zero (0x12)] test_CreateSubscription() (gas: 853041)
[PASS] test_ERC165() (gas: 13830)
[FAIL: panic: division or modulo by zero (0x12)] test_FirstStakeWithAllValue() (gas: 450186)
[FAIL: panic: division or modulo by zero (0x12)] test_FirstStakeWithPartialValue() (gas: 1028376)
[FAIL: panic: division or modulo by zero (0x12)] test_NonFirstStakeWithAllValue() (gas: 464068)
[FAIL: panic: division or modulo by zero (0x12)] test_NonFirstStakeWithPartialValue() (gas: 1042340)
[PASS] test_RevertWhenCreateRedemptionWithInvalidPoolId() (gas: 83776)
[PASS] test_RevertWhenSetOpenFundMarketByNonAdmin() (gas: 18214)
[PASS] test_RevertWhenSetOpenFundMarketWithInvalidAddress() (gas: 16588)
[PASS] test_RevertWhenSetSolvBTCMultiAssetPoolByNonAdmin() (gas: 18283)
[PASS] test_RevertWhenSetSolvBTCMultiAssetPoolWithInvalidAddress() (gas: 16646)
[PASS] test_RevertWhenSubscriberNotInWhitelist() (gas: 120746)
[PASS] test_RouterInitialStatus() (gas: 25483)
[PASS] test_SetOpenFundMarket() (gas: 25978)
[PASS] test_SetSolvBTCMultiAssetPool() (gas: 26017)
[FAIL: panic: division or modulo by zero (0x12)] test_SolvBTC_OnERC3525Received_FirstStake() (gas: 926756)
[FAIL: panic: division or modulo by zero (0x12)] test_SolvBTC_OnERC3525Received_NotFirstStake() (gas: 903269)
[FAIL: panic: division or modulo by zero (0x12)] test_SolvBTC_OnERC721Received_FirstStake() (gas: 400084)
[FAIL: panic: division or modulo by zero (0x12)] test_SolvBTC_OnERC721Received_NotFirstStake() (gas: 365831)
[FAIL: panic: division or modulo by zero (0x12)] test_UnstakeWhenGivenSftId() (gas: 1004964)
[FAIL: panic: division or modulo by zero (0x12)] test_UnstakeWhenNotGivenSftId() (gas: 1005052)
Suite result: FAILED. 10 passed; 13 failed; 0 skipped; finished in 166.07ms (30.26ms CPU time)
```

Ran 4 tests for test/XSolvBTCOracle.t.sol:XSolvBTCOracleTest

```
[FAIL: backend: failed while inspecting] test_GetLatestUpdatedAt() (gas: 0)
[FAIL: backend: failed while inspecting] test_RevertWhenGetNavByInvalidXSolvBTC() (gas: 0)
[FAIL: backend: failed while inspecting] test_RevertWhenSetNavByNonAdmin() (gas: 0)
[FAIL: backend: failed while inspecting] test_SetNav() (gas: 0)
Suite result: FAILED. 0 passed; 4 failed; 0 skipped; finished in 6.80s (5.88ms CPU time)
```

Ran 10 tests for test/XSolvBTC.t.sol:xSolvBTCTest

```
[FAIL: backend: failed while inspecting] test_SolvBTCRouterV2_depositSolvBTCToXSolvBTC() (gas: 0)
[FAIL: backend: failed while inspecting] test_SolvBTCRouterV2_depositWBCToSolvBTCBera() (gas: 0)
[FAIL: backend: failed while inspecting] test_XSolvBTCPool_depositSolvBTCToXSolvBTC() (gas: 0)
[FAIL: backend: failed while inspecting] test_XSolvBTCPool_withdrawSolvBTCFromXSolvBTC() (gas: 0)
[FAIL: backend: failed while inspecting] test_xSolvBTCPool_RevertWhenDepositAllowedIsFalse() (gas: 0)
[FAIL: backend: failed while inspecting] test_xSolvBTCPool_RevertWhenSetFeeRecipientByNonAdmin() (gas: 0)
[FAIL: backend: failed while inspecting] test_xSolvBTCPool_RevertWhenSetWithdrawFeeRateByNonAdmin() (gas: 0)
[FAIL: backend: failed while inspecting] test_xSolvBTCPool_setDepositAllowedOnlyAdmin() (gas: 0)
[FAIL: backend: failed while inspecting] test_xSolvBTCPool_setFeeRecipientOnlyAdmin() (gas: 0)
[FAIL: backend: failed while inspecting] test_xSolvBTCPool_setWithdrawFeeRateOnlyAdmin() (gas: 0)
Suite result: FAILED. 0 passed; 10 failed; 0 skipped; finished in 12.07s (226.62µs CPU time)
```

Ran 18 test suites in 12.08s (20.70s CPU time): 115 tests passed, 49 failed, 0 skipped (164 total tests)

Failing tests:

Encountered 1 failing test in test/NativeDeposit.t.sol:NativeDepositTest  
[FAIL: EvmError: Revert] test\_NativeDeposit() (gas: 7751)

Encountered 1 failing test in test/SolvBTC.t.sol:SolvBTCTest

[FAIL: panic: division or modulo by zero (0x12)] test\_SweepEmptySftIds() (gas: 3998437)

Encountered 10 failing tests in test/SolvBTCMultiAssetPool.t.sol:SolvBTCMultiAssetPoolTest

[FAIL: panic: division or modulo by zero (0x12)] test\_FullDepositWhenHoldingValueSftIdIsNotZero() (gas:



```
442992)
[FAIL: revert: ERC3525: invalid token ID] test_FullDepositWhenHoldingValueSftIdIsZero() (gas: 212496)
[FAIL: revert: ERC3525: invalid token ID] test_PartialDepositWhenHoldingValueSftIdIsZero() (gas: 212576)
[FAIL: Error != expected error: SolvBTCMultiAssetPool: withdraw amount cannot be 0 !=
SolvBTCMultiAssetPool: insufficient balance] test_RevertWhenWithdrawButPoolHoldingValueSftIdIsZero()
(gas: 207242)
[FAIL: Error != expected error: SolvBTCMultiAssetPool: withdraw amount cannot be 0 !=
SolvBTCMultiAssetPool: sft slot not allowed] test_RevertWhenWithdrawToInvalidSftSlot() (gas: 41582)
[FAIL: Error != expected error: SolvBTCMultiAssetPool: withdraw amount cannot be 0 !=
SolvBTCMultiAssetPool: caller is not sft owner] test_RevertWhenWithdrawToSftIdNotOwned() (gas: 207309)
[FAIL: Error != expected error: ERC3525: invalid token ID != SolvBTCMultiAssetPool: slot not matched]
test_RevertWhenWithdrawWithMismatchedSpecifiedSlot() (gas: 498284)
[FAIL: revert: SolvBTCMultiAssetPool: withdraw amount cannot be 0]
test_WithdrawForThoseWhoDepositBeforeUpgrade() (gas: 216716)
[FAIL: panic: division or modulo by zero (0x12)] test_WithdrawSlotAAfterDepositSlotA() (gas: 427065)
[FAIL: revert: ERC3525: invalid token ID] test_WithdrawSlotBAfterDepositSlotA() (gas: 453444)

Encountered 13 failing tests in test/SolvBTCRouter.t.sol:SolvBTCRouterTest
[FAIL: panic: division or modulo by zero (0x12)] test_CancelRedemption() (gas: 835383)
[FAIL: panic: division or modulo by zero (0x12)] test_CreateRedemption() (gas: 835424)
[FAIL: panic: division or modulo by zero (0x12)] test_CreateSubscription() (gas: 853041)
[FAIL: panic: division or modulo by zero (0x12)] test_FirstStakeWithAllValue() (gas: 450186)
[FAIL: panic: division or modulo by zero (0x12)] test_FirstStakeWithPartialValue() (gas: 1028376)
[FAIL: panic: division or modulo by zero (0x12)] test_NonFirstStakeWithAllValue() (gas: 464068)
[FAIL: panic: division or modulo by zero (0x12)] test_NonFirstStakeWithPartialValue() (gas: 1042340)
[FAIL: panic: division or modulo by zero (0x12)] test_SolvBTC_OnERC3525Received_FirstStake() (gas:
926756)
[FAIL: panic: division or modulo by zero (0x12)] test_SolvBTC_OnERC3525Received_NotFirstStake() (gas:
903269)
[FAIL: panic: division or modulo by zero (0x12)] test_SolvBTC_OnERC721Received_FirstStake() (gas: 400084)
[FAIL: panic: division or modulo by zero (0x12)] test_SolvBTC_OnERC721Received_NotFirstStake() (gas:
365831)
[FAIL: panic: division or modulo by zero (0x12)] test_UnstakeWhenGivenSftId() (gas: 1004964)
[FAIL: panic: division or modulo by zero (0x12)] test_UnstakeWhenNotGivenSftId() (gas: 1005052)

Encountered 4 failing tests in test/SolvBTCRouterV2.t.sol:SolvBTCRouterV2Test
[FAIL: EvmError: Revert] test_CancelWithdrawRequest() (gas: 5653)
[FAIL: EvmError: Revert] test_Deposit_WBTC_for_SOLVBTC() (gas: 5651)
[FAIL: EvmError: Revert] test_Deposit_WBTC_for_SOLVBTCBBN() (gas: 5607)
[FAIL: EvmError: Revert] test_WithdrawRequest() (gas: 5652)

Encountered 1 failing test in test/SolvBTCV2_1.t.sol:SolvBTCV2_1Test
[FAIL: setup failed: EvmError: Revert] setUp() (gas: 0)

Encountered 1 failing test in test/SolvBTCV3.t.sol:SolvBTCV3Test
[FAIL: setup failed: EvmError: Revert] setUp() (gas: 0)

Encountered 1 failing test in test/SolvBTCV3_1.t.sol:SolvBTCV3_1Test
[FAIL: setup failed: EvmError: Revert] setUp() (gas: 0)

Encountered 1 failing test in test/SolvBTCYieldTokenV2_1.t.sol:SolvBTCYieldTokenV2Test
[FAIL: setup failed: EvmError: Revert] setUp() (gas: 0)

Encountered 1 failing test in test/SolvBTCYieldTokenV3.t.sol:SolvBTCYieldTokenV3Test
[FAIL: setup failed: EvmError: Revert] setUp() (gas: 0)

Encountered 1 failing test in test/SolvBTCYieldTokenV3_1.t.sol:SolvBTCYieldTokenV3_1Test
[FAIL: setup failed: EvmError: Revert] setUp() (gas: 0)

Encountered 10 failing tests in test/XSolvBTC.t.sol:xSolvBTCTest
[FAIL: backend: failed while inspecting] test_SolvBTCRouterV2_depositSolvBTCtoXSolvBTC() (gas: 0)
[FAIL: backend: failed while inspecting] test_SolvBTCRouterV2_depositWBCToSolvBTCBera() (gas: 0)
[FAIL: backend: failed while inspecting] test_XSolvBTCPool_depositSolvBTCtoXSolvBTC() (gas: 0)
[FAIL: backend: failed while inspecting] test_XSolvBTCPool_withdrawSolvBTCFromXSolvBTC() (gas: 0)
[FAIL: backend: failed while inspecting] test_xSolvBTCPool_RevertWhenDepositAllowedIsFalse() (gas: 0)
[FAIL: backend: failed while inspecting] test_xSolvBTCPool_RevertWhenSetFeeRecipientByNonAdmin() (gas: 0)
[FAIL: backend: failed while inspecting] test_xSolvBTCPool_RevertWhenSetWithdrawFeeRateByNonAdmin() (gas:
0)
[FAIL: backend: failed while inspecting] test_xSolvBTCPool_setDepositAllowedOnlyAdmin() (gas: 0)
[FAIL: backend: failed while inspecting] test_xSolvBTCPool_setFeeRecipientOnlyAdmin() (gas: 0)
[FAIL: backend: failed while inspecting] test_xSolvBTCPool_setWithdrawFeeRateOnlyAdmin() (gas: 0)
```

```
Encountered 4 failing tests in test/XSolvBTCOracle.t.sol:XSolvBTCOracleTest
[FAIL: backend: failed while inspecting] test_GetLatestUpdatedAt() (gas: 0)
[FAIL: backend: failed while inspecting] test_RevertWhenGetNavByInvalidXSolvBTC() (gas: 0)
[FAIL: backend: failed while inspecting] test_RevertWhenSetNavByNonAdmin() (gas: 0)
[FAIL: backend: failed while inspecting] test_SetNav() (gas: 0)
```

Encountered a total of 49 failing tests, 115 tests succeeded

## Code Coverage

The test coverage failed to execute with failing tests.

**Fix Review Update:** The tests coverage could not be run due to failing tests.

## Changelog

- 2025-05-29 - Initial report
- 2025-06-24 - Final Report

## About Quantstamp

Quantstamp is a global leader in blockchain security. Founded in 2017, Quantstamp's mission is to securely onboard the next billion users to Web3 through its best-in-class Web3 security products and services.

Quantstamp's team consists of cybersecurity experts hailing from globally recognized organizations including Microsoft, AWS, BMW, Meta, and the Ethereum Foundation. Quantstamp engineers hold PhDs or advanced computer science degrees, with decades of combined experience in formal verification, static analysis, blockchain audits, penetration testing, and original leading-edge research.

To date, Quantstamp has performed more than 500 audits and secured over \$200 billion in digital asset risk from hackers. Quantstamp has worked with a diverse range of customers, including startups, category leaders and financial institutions. Brands that Quantstamp has worked with include Ethereum 2.0, Binance, Visa, PayPal, Polygon, Avalanche, Curve, Solana, Compound, Lido, MakerDAO, Arbitrum, OpenSea and the World Economic Forum.

Quantstamp's collaborations and partnerships showcase our commitment to world-class research, development and security. We're honored to work with some of the top names in the industry and proud to secure the future of web3.

Notable Collaborations & Customers:

- Blockchains: Ethereum 2.0, Near, Flow, Avalanche, Solana, Cardano, Binance Smart Chain, Hedera Hashgraph, Tezos
- DeFi: Curve, Compound, Maker, Lido, Polygon, Arbitrum, SushiSwap
- NFT: OpenSea, Parallel, Dapper Labs, Decentraland, Sandbox, Axie Infinity, Illuvium, NBA Top Shot, Zora
- Academic institutions: National University of Singapore, MIT

### Timeliness of content

The content contained in the report is current as of the date appearing on the report and is subject to change without notice, unless indicated otherwise by Quantstamp; however, Quantstamp does not guarantee or warrant the accuracy, timeliness, or completeness of any report you access using the internet or other means, and assumes no obligation to update any information following publication or other making available of the report to you by Quantstamp.

### Notice of confidentiality

This report, including the content, data, and underlying methodologies, are subject to the confidentiality and feedback provisions in your agreement with Quantstamp. These materials are not to be disclosed, extracted, copied, or distributed except to the extent expressly authorized by Quantstamp.

### Links to other websites

You may, through hypertext or other computer links, gain access to web sites operated by persons other than Quantstamp. Such hyperlinks are provided for your reference and convenience only, and are the exclusive responsibility of such web sites' owners. You agree that Quantstamp are not responsible for the content or operation of such web sites, and that Quantstamp shall have no liability to you or any other person or entity for the use of third-party web sites. Except as described below, a hyperlink from this web site to another web site does not imply or mean that Quantstamp endorses the content on that web site or the operator or operations of that site. You are solely responsible for determining the extent to which you may use any content at any other web sites to which you link from the report. Quantstamp assumes no responsibility for the use of third-party software on any website and shall have no liability whatsoever to any person or entity for the accuracy or completeness of any output generated by such software.

### Disclaimer

The review and this report are provided on an as-is, where-is, and as-available basis. To the fullest extent permitted by law, Quantstamp disclaims all warranties, expressed implied, in connection with this report, its content, and the related services and products and your use



thereof, including, without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement. You agree that access and/or use of the report and other results of the review, including but not limited to any associated services, products, protocols, platforms, content, and materials, will be at your sole risk. FOR AVOIDANCE OF DOUBT, THE REPORT, ITS CONTENT, ACCESS, AND/OR USAGE THEREOF, INCLUDING ANY ASSOCIATED SERVICES OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, INVESTMENT, TAX, LEGAL, REGULATORY, OR OTHER ADVICE. This report is based on the scope of materials and documentation provided for a limited review at the time provided. You acknowledge that Blockchain technology remains under development and is subject to unknown risks and flaws and, as such, the report may not be complete or inclusive of all vulnerabilities. The review is limited to the materials identified in the report and does not extend to the compiler layer, or any other areas beyond the programming language, or programming aspects that could present security risks. The report does not indicate the endorsement by Quantstamp of any particular project or team, nor guarantee its security, and may not be represented as such. No third party is entitled to rely on the report in any way, including for the purpose of making any decisions to buy or sell a product, service or any other asset. Quantstamp does not warrant, endorse, guarantee, or assume responsibility for any product or service advertised or offered by a third party, or any open source or third-party software, code, libraries, materials, or information to, called by, referenced by or accessible through the report, its content, or any related services and products, any hyperlinked websites, or any other websites or mobile applications, and we will not be a party to or in any way be responsible for monitoring any transaction between you and any third party. As with the purchase or use of a product or service through any medium or in any environment, you should use your best judgment and exercise caution where appropriate.

