coinspect

You build, we defend.

SATLAYER

Smart Contract Audit

SatLayer Core V2

April, 2025

SatLayer Core V2

Smart Contract Audit

Version: v250404 | Prepared for: SatLayer | April 2025

Security Assessment

1. Executive Summary

In **March, 2025**, **SatLayer** engaged Coinspect to perform a Smart Contract Audit of SatLayer's core.

The objective of the project was to evaluate the security of code refactoring and the inclusion of new features to the protocol's core smart contracts.

	✓ Solved	⚠ Caution Advised	✗ Resolution Pending
High	0	0	0
Medium	0	0	0
Low	1	0	0
No Risk	0	0	0
Total	**1**	**0**	**0**

During this assessment, Coinspect identified one low-risk issue related to the usage of non standard tokens in vaults.

2. Summary of Findings

This section provides a concise overview of all the findings in the report grouped by remediation status and sorted by estimated total risk.

2.3 Solved issues & recommendations

These issues have been fully fixed or represent recommendations that could improve the long-term security posture of the project.

Id	Title	Risk
SATR-01	Strategies using non-standard CW20 leave an unbacked portion of shares	Low

3. Scope

The scope was set to be the repository at https://github.com/satlayer/satlayer-bvs/tree/v0.5.0 at commit `03650d141f8b2633b2573b7959df042d409ab22a`.

With the following files in scope:

- `crates/bvs-vault-router`
- `crates/bvs-vault-base`
- `crates/bvs-vault-bank`
- `crates/bvs-vault-cw20`

Beyond the scope of this review, there are still several files on the repository that were part of the previous review's scope and have issues that should be addressed before advancing to the production phase.

4. Assessment

SatLayer is a Bitcoin Validated Service (BVS) solution built on Babylon, featuring interconnected contracts for staking, delegation, slashing, rewards, and validator selection.

The protocol involves three key roles/actors:

- **BTC Restakers**: deposit and stake Bitcoin, delegating it to Operators. If an Operator behaves maliciously, their staked BTC may be slashed. Restakers receive rewards in return.
- **BVS Developers**: launch Bitcoin Validated Services (BVS) using staked BTC, addressing security challenges for new services. They maintain both on-chain BVS contracts and off-chain operator software while managing reward distribution.
- **Operators**: deploy and run BVS, maintaining network connectivity, execution environments, and hardware. They earn a portion of rewards for providing these resources. Operators independently choose which BVS applications to support, with fees varying based on selection. SatLayer operates as a permissionless system, where market dynamics determine operator participation and BVS adoption.

These roles interact with the protocol thanks to a set of contracts that enable the before mentioned functionality.

This assessment covers the refactoring of the previous protocol version alongside the addition of new features and mitigations against previously reported attacks.

4.1 Security Assumptions

For this assessment, Coinspect made the following assumptions:

- Smart contracts are managed by honest accounts that do not act in bad faith.
- New strategies are going to be audited and reviewed before integrating them into the protocol.
- No other actor or party using/operating in the system should be considered trusted. All parties could be malicious or dishonest.

4.2 New features

The changes and features reviewed in this audit are the following:

- **Vault Router**: Acts as an entry/exit point for all vaults, designed to support delegation/slashing routing in the future, currently minimally implemented, planned to be integrated in Phase 2.

- **Vault Base**: Building blocks that provide base utilities to implement both Bank and CW20 vaults. These utilities implement abstraction code for `ERC4626`-like functionality including protection against inflation attacks and accounting of shares.

- **Vault Bank**: Vault implementation that supports native `denom`. Accepts native `denom` currency and mints shares.

- **Vault CW20**: Vault implementation that supports CW20 tokens. Accepts CW20 and mints shares.

4.3 Changes with previous audit

- `contracts/strategy-manager` moved into `crates/bvs-vault-router`: shares accounting moved into vaults instead of managed by a single contract.
- `contracts/strategy-*` moved into `crates/bvs-vault-base`, `crates/bvs-vault-bank` and `crates/bvs-vault-cw20`
- `contracts/directory` and `contracts/delegation` merged into `crates/bvs-registry`: out of this review's scope
- `contracts/slash-manager` moved into `crates/bvs-slashing`: out of this review's scope
- `contracts/rewards-coordinator`: moved out of scope.
- `modules/*` Golang Code: out of scope.
- `examples/*` Code: out of scope.

4.4 Decentralization

The protocol requires to scrutinize the deployed strategies since they play a sensitive role in the system. Moreover, smart contracts rely on both the figures of an administrator (specified when instantiating smart contracts) and an owner (handled at each smart contract with setters).

The administrator is allowed to perform source code migrations and also resign the right to do so by clearing the administrator privileges. Not specifying an administrator when instantiating a contract will prevent future migrations turning the implementation immutable.

Smart contract owners have the privilege to manage supported vaults and trigger emergency pause at anytime.

4.5 Documentation and Testing

A set of tests is provided for the refactored and revamped smart contracts. Coinspect was able to test and evaluate multiple scenarios thanks to the clear and easy to understand test suite. Moreover, comments and documentation are included on important functions and variables easing the process of understanding their purpose and use cases.

5. Detailed Findings

SATR-01

Strategies using non-standard CW20 leave an unbacked portion of shares

Status

Solved

✓

Resolution

Fixed

Risk

Low

▼

🟥 □ □

Impact

Medium

Likelihood

Low

Location

```
satlayer-bvs/crates/bvs-vault-cw20/src/contract.rs:99
```

Description

Strategies using non-standard CW20 underlying tokens, such as those that charge a fee upon transfer, will mint more shares than the actual assets received by the contract. As a consequence, the internal accountancy will be broken and will not reflect the actual share to token ratio.

The calculation of the amount of shares granted to the recipient assumes that the vault receives all the assets specified by the function's parameter:

```
        let (virtual_offset, new_shares) = {
        let balance = token::query_balance(&deps.as_ref(), &env)?;
        let mut virtual_offset =
offset::VirtualOffset::load(&deps.as_ref(), balance)?;

        let new_shares = virtual_offset.assets_to_shares(assets)?;
        // Add shares to TOTAL_SHARES
        virtual_offset.checked_add_shares(deps.storage, new_shares)?;

        (virtual_offset, new_shares)
    };

    // CW20 Transfer of asset from info.sender to contract
    let transfer_msg = token::execute_transfer_from(
        deps.storage,
        &info.sender,
        &env.contract.address,
        msg.amount,
    )?;

    // Add shares to msg.recipient
    shares::add_shares(deps.storage, &msg.recipient, new_shares)?;
```

However, when using non standard CW20 such as one that diverts a percentage of each transfer, the amount of assets received by the vault is less. As a consequence, since users are granted with a share that is backed with more assets than those backed, the last user might not be able to withdraw all their position.

Coinspect considers the likelihood to be low since it requires deploying a strategy that uses a non-standard `CW20` as underlying token. The impact is medium since it depends on the imbalance made on each deposit.

Recommendation

Use effective received balances to calculate the amount of issues shares. Alternatively, ensure that no vault is deployed using a non-standard CW20 token.

Status

Addressed at `5f907e36bdf2320f6456aa23fa4224682be24b9d` by including documentation related to the supported token types.

6. Disclaimer

The contents of this report are provided "as is" without warranty of any kind. Coinspect is not responsible for any consequences of using the information contained herein.

This report represents a point-in-time and time-boxed evaluation conducted within a specific timeframe and scope agreed upon with the client. The assessment's findings and recommendations are based on the information, source code, and systems access provided by the client during the review period.

The assessment's findings should not be considered an exhaustive list of all potential security issues. This report does not cover out-of-scope components that may interact with the analyzed system, nor does it assess the operational security of the organization that developed and deployed the system.

This report does not imply ongoing security monitoring or guaranteeing the current security status of the assessed system. Due to the dynamic nature of information security threats, new vulnerabilities may emerge after the assessment period.

This report should not be considered an endorsement or disapproval of any project or team. It does not provide investment advice and should not be used to make investment decisions.