DEDAUB

DEDAUB.COM

SatLayer

SLAY ERC-20

Smart Contract Security Assessment

June 16, 2025

ABSTRACT

Dedaub was commissioned to perform a security audit of SatLayer's SLAY token. The token is a standard ERC-20 implementation that utilizes OpenZeppelin's Solidity libraries for both core functionality and upgradeability. No security-related issues were identified.

SETTING & CAVEATS

This audit report mainly covers the contracts of the **at-the-time private** repository https://github.com/satlayer/token at commit b2c2e140fc649ef0ffea6b90cb948c87d5f83281.

2 auditors worked on the following contracts for 0.5 days:

```
contract
├── script
│   └── SLAYToken.deployment.sol
├── src
│   └── SLAYToken.sol
└── test
    └── SLAYToken.t.sol
```

The audit's main target is security threats, i.e., what the community understanding would likely call "hacking", rather than the regular use of the protocol. Functional correctness (i.e. issues in "regular use") is a secondary consideration. Typically it can only be covered if we are provided with unambiguous (i.e. full-detail) specifications of what is the expected, correct behavior. In terms of functional correctness, we often trusted the code's calculations and interactions, in the absence of any other specification. Functional correctness relative to low-level calculations (including units, scaling and quantities returned from external protocols) is generally most effectively done through thorough testing rather than human auditing.

VULNERABILITIES & FUNCTIONAL ISSUES

This section details issues affecting the functionality of the contract. Dedaub generally categorizes issues according to the following severities, but may also take other considerations into account such as impact or difficulty in exploitation:

Category	Description
CRITICAL	Can be profitably exploited by any knowledgeable third-party attacker to drain a portion of the system's or users' funds OR the contract does not function as intended and severe loss of funds may result.
HIGH	Third-party attackers or faulty functionality may block the system or cause the system or users to lose funds. Important system invariants can be violated.
MEDIUM	Examples: • User or system funds can be lost when third-party systems misbehave. • DoS, under specific conditions. • Part of the functionality becomes unusable due to a programming error.
LOW	Examples: • Breaking important system invariants but without apparent consequences. • Buggy functionality for trusted users where a workaround exists. • Security issues which may manifest when the system evolves.

Issue resolution includes "dismissed" or "acknowledged" but no action taken, by the client, or "resolved", per the auditors.

CRITICAL SEVERITY:

[No critical severity issues]

HIGH SEVERITY:

[No high severity issues]

MEDIUM SEVERITY:

[No medium severity issues]

LOW SEVERITY:

[No medium severity issues]

CENTRALIZATION CONSIDERATIONS:

It is often desirable for DeFi protocols to assume no trust in a central authority, including the protocol's owner. Even if the owner is reputable, users are more likely to engage with a protocol that guarantees no catastrophic failure even in the case the owner gets hacked/compromised. We list considerations of this kind below. (These issues should be considered in the context of usage/deployment, as they are not uncommon. Several high-profile, high-value protocols have significant centralization threats.)

ID	Description	STATUS
N1	Some entities are considered trusted	**INFO**
The SLAY token is a UUPS upgradeable contract, with the owner having the power to upgrade its implementation at any time		

SLAYToken::_authorizeUpgrade:41

```
function _authorizeUpgrade(address newImplementation) internal override onlyOwner
{}
```

OTHER / ADVISORY ISSUES:

This section details issues that are not thought to directly affect the functionality of the project, but we recommend considering them.

ID	Description	STATUS
A1	Compiler bugs	**INFO**

The code has a floating Solidity `^0.8.22` version. We recommend a fixed compiler version for deployment, i.e., no floating pragmas, to remove all ambiguity.
Version `0.8.22`, in particular, has some known bugs, which we do not believe affect the correctness of the contracts.

DISCLAIMER

The audited contracts have been analyzed using automated techniques and extensive human inspection in accordance with state-of-the-art practices as of the date of this report. The audit makes no statements or warranties on the security of the code. On its own, it cannot be considered a sufficient assessment of the correctness of the contract. While we have conducted an analysis to the best of our ability, it is our recommendation for high-value contracts to commission several independent audits, a public bug bounty program, as well as continuous security auditing and monitoring through Dedaub Security Suite.

ABOUT DEDAUB

Dedaub offers significant security expertise combined with cutting-edge program analysis technology to secure some of the most prominent protocols in DeFi. The founders, as well as many of Dedaub's auditors, have a strong academic research background together with a real-world hacker mentality to secure code. Protocol blockchain developers hire us for our foundational analysis tools and deep expertise in program analysis, reverse engineering, DeFi exploits, cryptography and financial mathematics.