PALADIN
BLOCKCHAIN SECURITY

Smart Contract Security Assessment

Final Report

For LayerZero (SolvBTC)

09 December 2024

paladinsec.co

info@paladinsec.co

Table of Contents

Disclaimer

Paladin Blockchain Security ("Paladin") has conducted an independent audit to verify the integrity of and highlight any vulnerabilities or errors, intentional or unintentional, that may be present in the codes that were provided for the scope of this audit. This audit report does not constitute agreement, acceptance or advocation for the Project that was audited, and users relying on this audit report should not consider this as having any merit for financial advice in any shape, form or nature. The contracts audited do not account for any economic developments that may be pursued by the Project in question, and that the veracity of the findings thus presented in this report relate solely to the proficiency, competence, aptitude and discretion of our independent auditors, who make no guarantees nor assurance that the contracts are completely free of exploits, bugs, vulnerabilities or deprecation of technologies. Further, this audit report shall not be disclosed nor transmitted to any persons or parties on any objective, goal or justification without due written assent, acquiescence or approval by Paladin.

All information provided in this report does not constitute financial or investment advice, nor should it be used to signal that any persons reading this report should invest their funds without sufficient individual due diligence regardless of the findings presented in this report. Information is provided 'as is', and Paladin is under no covenant to the completeness, accuracy or solidity of the contracts audited. In no event will Paladin or its partners, employees, agents or parties related to the provision of this audit report be liable to any parties for, or lack thereof, decisions and/or actions with regards to the information provided in this audit report.

Cryptocurrencies and any technologies by extension directly or indirectly related to cryptocurrencies are highly volatile and speculative by nature. All reasonable due diligence and safeguards may yet be insufficient, and users should exercise considerable caution when participating in any shape or form in this nascent industry.

The audit report has made all reasonable attempts to provide clear and articulate recommendations to the Project team with respect to the rectification, amendment and/or revision of any highlighted issues, vulnerabilities or exploits within the contracts provided. It is the sole responsibility of the Project team to sufficiently test and perform checks, ensuring that the contracts are functioning as intended, specifically that the functions therein contained within said contracts have the desired intended effects, functionalities and outcomes of the Project team.

Paladin retains the right to re-use any and all knowledge and expertise gained during the audit process, including, but not limited to, vulnerabilities, bugs, or new attack vectors. Paladin is therefore allowed and expected to use this knowledge in subsequent audits and to inform any third party, who may or may not be our past or current clients, whose projects have similar vulnerabilities. Paladin is furthermore allowed to claim bug bounties from third-parties while doing so.

1 Overview

This report has been prepared for LayerZero's SolvBTC's contracts on the Ethereum network. Paladin provides a user-centred examination of the smart contracts to look for vulnerabilities, logic errors or other issues from both an internal and external perspective.

1.1 Summary

Project Name	LayerZero
URL	https://layerzero.network/
Platform	Ethereum
Language	Solidity
Preliminary Contracts	https://github.com/LayerZero-Labs/solvbtc-oft/tree/21e60d13e44a0359495e857e021a67720137c7c3/contracts
Resolution #1	https://github.com/LayerZero-Labs/solvbtc-oft/pull/1/commits/1cf67eff33a2f096aa1893d497dd03ff248e14e7

1.2 Contracts Assessed

Name	Contract	Live Code Match
`SolvBTCAdapter`	`Proxy: 0xB12979Ff302Ac903849948037A51792cF7186E8e` `Implementation: 0x2265A8E449eA5FF78ee4C115370E9B5957358a68`	MATCH

1.3 Findings Summary

Severity	Found	Resolved	Partially Resolved	Acknowledged (no change made)
High	0	-	-	-
Medium	0	-	-	-
Low	0	-	-	-
Informational	6	3	1	2
Total	**6**	**3**	**1**	**2**

Classification of Issues

Severity	Description
High	Exploits, vulnerabilities or errors that will certainly or probabilistically lead towards loss of funds, control, or impairment of the contract and its functions. Issues under this classification are recommended to be fixed with utmost urgency.
Medium	Bugs or issues with that may be subject to exploit, though their impact is somewhat limited. Issues under this classification are recommended to be fixed as soon as possible.
Low	Effects are minimal in isolation and do not pose a significant danger to the project or its users. Issues under this classification are recommended to be fixed nonetheless.
Informational	Consistency, syntax or style best practices. Generally pose a negligible level of risk, if any.

1.3.1 SolvBTCAdapter

ID	Severity	Summary	Status
01	INFO	Missing `_disableInitializers` in the constructor	RESOLVED
02	INFO	User is required to approve the adapter to use `send()`	ACKNOWLEDGED
03	INFO	`_credit` can fail if `_to` is `address(0)`	RESOLVED
04	INFO	If adapter receives native funds, they will get stuck	ACKNOWLEDGED
05	INFO	`SolvBTCAdapter` should call `__MintAndBurnOFTAdapterWithFeeUpgradeable_init`	RESOLVED
06	INFO	Typographical issues	PARTIAL

2 Findings

2.1 SolvBTCAdapter

`SolvBTCAdapter` is an extension of the `MintAndBurnOFTAdapter` that implements a rate limiter which limits the `_debit` and `_credit` operations based on the limits configured by the owner.

The contract relies on `MintAndBurnOFTAdapterWithFeeAndRateLimitUpgradeable`, `FeeUpgradable` and `RateLimiterUpgradable` abstract contracts that were previously audited.

The current audit points out only the integration of these with the `SolvBTC` and `SolvBTC.BBN` tokens present at the following contract addresses:
https://etherscan.io/address/0x9f2a9f488d82683e8a4a51a581fd34155d50d541#code
https://etherscan.io/address/0x8add70845fbb80564503879a4a25e9b38856528c#code

2.1.1 Privileged Functions

- `withdrawFees`
- `setDefaultFeeBps`
- `setFeeBps`
- `setRateLimits`
- `transferOwnership`
- `renounceOwnership`

2.1.2 Issues & Recommendations

Issue #01	Missing `_disableInitializers` in the constructor
Severity	INFORMATIONAL
Description	Within the proxy pattern, a contract's implementation is initialized by using a function that has initializer modifier. This pattern is inherited from the `Initializable` contract. To avoid the initialization of the implementation by random actors, a `_disableInitializers` call is required in any constructor of the main contract. https://docs.openzeppelin.com/upgrades-plugins/1.x/writing-upgradeable#initializing_the_implementation_contract
Recommendation	Consider adding the `_disableInitializers` call in the constructor.
Resolution	RESOLVED

Issue #02	User is required to approve the adapter to use send( )
Severity	INFORMATIONAL
Description	In the original implementation of the `MintAndBurnOFTAdapter`, the user does not need to approve the contract in order to use the `send( )` function. It has been changed so that the user is required to increase the allowance of the adapter in order for the tokens to be transferred and the fees to be deducted from this transferred amount before being burned. While this does not pose a risk, the difference might be a bit confusing for the users.
Recommendation	If it is desired to keep the same features of the `MintAndBurnOFTAdapter` in `MintAndBurnOFTAdapterWithFeeAndRateLimitUpgradeable`, then the `_debit` function can be modified as follows: burn the full amount from the caller and just mint the fee to the contract—this way, there is no need for user approvals. Additionally, `SolvBTC` does have a function with the following signature function `burn(address account_, uint256 value_)` under the `SOLVBTC_POOL_BURNER_ROLE` role, which means that the adapter would need to have two roles: `SOLVBTC_MINTER_ROLE` needed for `mint` and `SOLVBTC_POOL_BURNER_ROLE` needed for this special `burn` operation.
Resolution	ACKNOWLEDGED

Issue #03	`_credit` can fail if `_to` is `address(0)`
Severity	INFORMATIONAL
Description	In the `_credit` function, `innerToken` is minted to the `_to` address. SolvBTC reverts if minting to `address(0)` is attempted: https://github.com/LayerZero-Labs/solvbtc-oft/blob/21e60d13e44a0359495e857e021a67720137c7c3/contracts/solvbtc/SolvBTC.sol#L88
Recommendation	Consider replacing the `_to` address with `address(0xdead)` when it is equal to `address(0)`: `if (_to == address(0x0)) _to = address(0xdead);`
Resolution	RESOLVED

Issue #04	If adapter receives native funds, they will get stuck
Severity	INFORMATIONAL
Description	If the user specifies extra options with their message that causes the executor to send a `msg.value` when calling `lzReceive()`, these funds will get stuck since `lzReceive()` does not handle `msg.value` in any way and there is no withdraw function that the adapter owner can use to retrieve the funds.
Recommendation	Consider implementing a `withdraw` function for the OApps that are not supposed to handle `msg.value`.
Resolution	ACKNOWLEDGED The team stated: "A similar issue can be raised for many other OApps. We're not advising anyone to send `msg.value` to that contract."

Issue #05	**`SolvBTCAdapter` should call `__MintAndBurnOFTAdapterWithFeeUpgradeable_init`**
Severity	INFORMATIONAL
Description	`SolvBTCAdapter` calls `__OFTAdapter_init` during initialization. However, to maintain consistency and adhere to the initialization pattern of dependencies, it should instead call `__MintAndBurnOFTAdapterWithFeeUpgradeable_init`. While this issue does not pose a direct risk as `__MintAndBurnOFTAdapterWithFeeUpgradeable_init` internally calls `__OFTAdapter_init`, this disrupts the expected sequence of subsequent initialization calls.
Recommendation	Consider calling `__MintAndBurnOFTAdapterWithFeeUpgradeable_init` in `initialize` instead of `__OFTAdapter_init`.
Resolution	RESOLVED

Issue #06	Typographical issues
Severity	INFORMATIONAL

Description	Missing NatSpec comments for the `SolvBTCAdapter`.
	
	— `MintAndBurnOFTAdapterWithFeeAndRateLimitUpgradeable#L113`
	
	The comment is a bit misleading:
	
	Burns tokens from the sender's specified balance, but transfers the fee to the contract. However, the logic is that the full amount is transferred to the contract, then the amount without the fee is burned from the contract and the fee remains locked in the contract. If this logic is kept then the comment needs to be improved.
	
	— `MintAndBurnOFTAdapterWithFeeAndRateLimitUpgradeable#L135`
	
	The comment can be improved to specify that the fees include the dust resulting from the de-dust operation:
	`// @dev increment the total fees that can be withdrawn should be`
	`// @dev increment the total fees that can be withdrawn. Fees include the dust resulting from the de-dust operation.`

Recommendation	Consider fixing the typographical issues.
Resolution	PARTIALLY RESOLVED

PALADIN
BLOCKCHAIN SECURITY