PALADIN
BLOCKCHAIN SECURITY

Smart Contract Security Assessment

Final Report

For PolySage Finance

04 October 2021

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.

1 Overview

This report has been prepared for PolySage Finance on the Polygon 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	PolySage Finance
URL	https://polysage.finance/
Platform	Polygon
Language	Solidity

1.2 Contracts Assessed

Name	Contract	Live Code Match
SageToken	0x2ed945Dc703D85c80225d95ABDe41cdeE14e1992	MATCH
MasterChef	0x0451b4893e4a77E7Eec3B25E816ed7FFeA1EBA68	MATCH
Timelock	0xc8391D2ADF3969f2ecea18e69Af6Ca88abD452cb	MATCH

1.3 Findings Summary

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

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 WiseToken

ID	Severity	Summary	Status
01	LOW	mint function can be used to pre-mint large amounts of tokens before ownership is transferred to the Masterchef	RESOLVED

1.3.2 MasterChef

ID	Severity	Summary	Status
02	LOW	`mint` uses raw subtraction	ACKNOWLEDGED

1.3.3 Timelock

No issues found.

2 Findings

2.1 SageToken

The contract allows for Sage tokens to be minted when the `mint` function is called by the Owner, who at the time of deployment would be the deployer. However, ownership is generally transferred to the Masterchef via the `transferOwnership` function for emission rewards to be minted and distributed to users staking in the Masterchef. The `mint` function can be used to pre-mint tokens for various uses including injection of initial liquidity, token presale, airdrops, and others.

2.1.1 Token Overview

Address	0x2ed945Dc703D85c80225d95ABDe41cdeE14e1992
Token Supply	4,800
Decimal Places	18
Transfer Max Size	None
Transfer Min Size	None
Transfer Fees	None

2.1.2 Privileged Roles

The following functions can be called by the owner of the contract:

- `mint`

2.1.3 Issues & Recommendations

Issue #01	**`mint` function can be used to pre-mint large amounts of tokens before ownership is transferred to the Masterchef**
Severity	LOW SEVERITY
Description	The `mint` function could be used to pre-mint tokens for legitimate uses including, but not limited to, the injection of initial liquidity, token presale, or airdrops; however, this function may also be used to pre-mint tokens for dumping.
Recommendation	Consider being forthright if this `mint` function has been used by letting your community know how much was minted, where they are currently stored, if a vesting contract was used for token unlocking, and finally the purpose of the mints.
Resolution	RESOLVED Ownership of the contract has been transferred to the Masterchef.

2.2 MasterChef

The PolySage Masterchef contract is a perfect fork of PolyWise Finance's Masterchef contract, which was previously audited by Paladin. As such, it is a secure Masterchef contract, and we commend PolySage on forking an audited, proven Masterchef. Deposit fees have an upper limit of 4%, transfer tax tokens are properly accounted for, and the notorious `migrator` function has also been removed.

A notable feature of this Masterchef is that the maximum token supply of 4,800 tokens is enforced in the `updatePool` function via the try/catch statement.

2.2.1 Privileged Roles

The following functions can be called by the owner of the contract:

- `add`
- `set`
- `setDevAddress`
- `setFeeAddress`
- `updateEmissionRate`
- `updateStartBlock`

2.2.2 Issues & Recommendations

Issue #02	`mint` uses raw subtraction
Severity	LOW SEVERITY
Location	Line 1235 `_amount = pool.lpToken.balanceOf(address(this)) - balanceBefore;`
Description	Despite the risk of underflows being low in this contract, the use of raw subtraction may cause underflow issues in certain edge cases as the contract is using Solidity version 0.6.12.
Recommendation	Consider either using Solidity versions 0.8.0 or higher, or implementing SafeMath's sub rather than using raw subtraction.
Resolution	ACKNOWLEDGED

2.3 Timelock

The Timelock contract is a clean fork of Compound Finance's timelock. This is the most common contract used in DeFi to time lock governance access and is thus compatible with most third-party tools.

Parameter	Value	Description
Delay	4 hours	The `delay` indicates the time the administrator has to wait after queuing a transaction to execute it.
Minimum Delay	4 hours	The `minDelay` indicates the lowest value that the `delay` can minimally be set. Sometimes, projects will queue a transaction that sets the `delay` to zero with the hope that nobody notices it. However, because of the minimum delay parameter, the value of `delay` can never be lower than that of the `minDelay` value. Note that the administrator could still queue a transaction to simply transfer the ownership back to their own account so it is still important to inspect every transaction carefully.
Grace Period	14 days	After the delay has expired after queueing a transaction, the administrator can only execute it within the grace period. This is to prevent them from hiding a malicious transaction among much earlier transactions, hoping that it goes unnoticed or buried, which can be executed in the future.

2.3.1 Issues & Recommendations

No issues found.


PALADIN
BLOCKCHAIN SECURITY