PALADIN
BLOCKCHAIN SECURITY

Smart Contract Security Assessment

Final Report

For AvalancheFarm Finance

08 November 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 Avalanche Farm on the Fantom 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	Avalanche Farm
URL	
Platform	Fantom
Language	Solidity

1.2 Contracts Assessed

Name	Contract	Live Code Match
SpadeToken	0xdF5d2365f09943F6080f9DcE0E0D741B93718A75	✓ MATCH
MasterChef	0xdeff285d0115729d0915a84deadf1fce2a41d1e5	✓ MATCH

1.3 Findings Summary

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

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	INFO	Total token supply might not be minted due to try and catch pattern	ACKNOWLEDGED
03	INFO	`MAX_EMISSION_RATE` can be declared as a `constant`	ACKNOWLEDGED

2 Findings

2.1 SpadeToken

The SPADE token is a simple ERC-20 token which will be used as the main reward token for the Masterchef. It allows for SPADE tokens to be minted when the mint function is called by the owner of the contract, which at the time of deployment would be the SPADE team. Users should therefore carefully inspect that the ownership of the contract has been transferred to the Masterchef. The token has a maximum supply of 10,000.

2.1.1 Token Overview

Address	0xdF5d2365f09943F6080f9DcE0E0D741B93718A75
Token Supply	10,000
Decimal Places	18
Transfer Max Size	No maximum
Transfer Min Size	No minimum
Transfer Fees	None
Pre-mints	50

2.1.2 Privileged Roles

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

- `mint`
- `renounceOwnership`
- `transferOwnership`

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 and dump tokens when the token contract has been deployed but before ownership is set to the Masterchef contract. This risk is prevalent amongst less-reputable projects, and any pre-mints can be prominently seen on the Blockchain.
Recommendation	Consider being forthright if this `mint` function is to be 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 50 tokens were pre-minted and ownership has been transferred to the Masterchef.

2.2 MasterChef

The Spade Masterchef contract was forked from PolyBeta, which was previously audited by Paladin. As such, it is a battle-tested and secure Masterchef. Notably, there are no hard risk functionalities within the contract. Deposit fees have an upper limit of 4%, and the upper limit of 20,000 tokens is enforced via the try/catch implementation in the `updatePool` function.

As Spade Masterchef is deployed on the AVAX network, the team has used block timestamp instead of block height for the calculation of rewards.

2.2.1 Privileged Roles

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

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

2.2.2 Issues & Recommendations

Issue #02	Total token supply might not be minted due to try and catch pattern
Severity	● INFORMATIONAL

Description	As there is a MAXCAPSUPPLY for the Spade token, minting the reward and causing the max cap to exceed would result in a revert.
	
	`require(MAXCAP.add(amount) <= MAXCAPSUPPLY, "Max supply reached");`
	
	To prevent this, the following try and catch pattern is done in `updatePool`:
	
	L1205~
	`try spade.mint(devaddr, spadeReward.div(10)) {`
	`} catch (bytes memory reason) {`
	`    spadeReward= 0;`
	`    emit SpadeMintError(reason);`
	`}`
	
	`try spade.mint(address(this), spadeReward) {`
	`} catch (bytes memory reason) {`
	`    spadeReward= 0;`
	`    emit SpadeMintError(reason);`
	`}`
	
	In the case where `MAXCAP + amount` does exceed `MAXCAPSUPPLY`, the mint will not be done. This means that the token supply could be capped at an amount slightly lower than `MAXCAPSUPPLY`.

Recommendation	Consider minting the difference between `MAXCAPSUPPLY - MAXCAP`, if any.
Resolution	● ACKNOWLEDGED

Issue #03	`MAX_EMISSION_RATE` can be declared as a `constant`
Severity	● INFORMATIONAL
Description	As `MAX_EMISSION_RATE` is only declared once as a state variable and never changed, it can be declared as a constant for gas optimization.
Recommendation	Consider adding the `constant` keyword when declaring `MAX_EMISSION_RATE` .
Resolution	● ACKNOWLEDGED

PALADIN

BLOCKCHAIN SECURITY