PALADIN
BLOCKCHAIN SECURITY

Smart Contract Security Assessment

Final Report

For Seasonal Tokens Polygon Farm

20 October 2022

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 full rights over all intellectual property (including expertise and new attack or exploit vectors) discovered during the audit process. Paladin is therefore allowed and expected to re-use this knowledge in subsequent audits and to inform existing projects that may have similar vulnerabilities. Paladin may, at its discretion, claim bug bounties from third-parties while doing so.

1 Overview

This report has been prepared for Seasonal Tokens Polygon Farm 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	Seasonal Tokens Polygon Farm
URL	https://seasonaltokens.org/
Network	Polygon
Language	Solidity

1.2 Contracts Assessed

Name	Contract	Live Code Match
`SeasonalTokenFarm`	`0x27114Bb43Ca5B3fc13bf51284aa036Ed5869B371`	✓ MATCH

Note: The deployed contract matches the audited code with a small change on line 88 as the client stated that the rewards for the tokens had started on 5 Jan 2021.

1.3 Findings Summary

Severity	Found	Resolved	Partially Resolved	Acknowledged (no change made)
High	0	-	-	-
Medium	0	-	-	-
Low	1	1	-	-
Informational	4	3	-	1
Total	**5**	**4**	**-**	**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 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 SeasonalTokenFarm

ID	Severity	Summary	Status
01	LOW	UniswapV3 fees of transferred positions cannot be collected	RESOLVED
02	INFO	`removeTokenFromListOfOwnedTokens` can be simplified	RESOLVED
03	INFO	`receiveSeasonalTokens` can be frontrun	ACKNOWLEDGED
04	INFO	Safeguard missing in the constructor	RESOLVED
05	INFO	Gas optimizations and typographical errors	RESOLVED

2 Findings

2.1 SeasonalTokenFarm

`SeasonalTokenFarm` is a staking contract which allows investors to stake NFT-like liquidity tokens. Although the type of token was not included within the scope of this audit, we assume it to be ERC721 wrapped UniswapV3 tokens.

Each LP is rewarded with the four seasonal tokens. However, the allocation points of the LPs vary over time and are automatically adjusted every 9 months according to a repeating schedule over four 9 month periods. Currently, the contract has no notion of governance and runs off donations.

2.1.1 Issues & Recommendations

Issue #01	UniswapV3 fees of transferred positions cannot be collected
Severity	LOW SEVERITY
Description	The `SeasonalTokenFarm` contract receives UniswapV3 liquidity positions from the users. These positions still collect fees on UniswapV3 but these fees can not be collected as there is no way for the users to call the `collect` function on the position manager.
Recommendation	Consider implementing a proxy function within the contract that calls the `nonfungiblePositionManager` to claim the fees of the positions and send them to the owners.
Resolution	RESOLVED The client has indicated that these fees will be negligible and after some discussion we agree that adding claim logic unnecessarily complicates the contract .

Issue #02	`removeTokenFromListOfOwnedTokens` can be simplified
Severity	INFORMATIONAL
Description	`removeTokenFromListOfOwnedTokens` removes a token from the `tokenOfOwnerByIndex` mapping. This mapping uses an array to keep track of UniswapV3 tokens sent by an owner. The logic of this function can be improved by using the `EnumerableSet` library from OpenZeppelin which allows elements to be easily added and removed from a set.
Recommendation	Consider using the `EnumerableSet` library from OpenZeppelin instead of an array.
Resolution	RESOLVED

Issue #03	`receiveSeasonalTokens` can be frontrun
Severity	INFORMATIONAL
Description	`receiveSeasonalTokens` is used to deposit Seasonal Tokens. This function can be frontrun to extract value if a large staker is depositing a huge amount.
Recommendation	Consider allocating the donations over time. An example can be seen in the Synthetix approach.
Resolution	ACKNOWLEDGED

Issue #04	Safeguard missing in the constructor
Severity	INFORMATIONAL
Description	The `_startTime` parameter within the constructor is missing validation to avoid being mistakenly set in the future.
Recommendation	Consider adding a check within the constructor for `_startTime >= block.timestamp`.
Resolution	RESOLVED

Issue #05	Gas optimizations and typographical errors
Severity	INFORMATIONAL

Description	We have consolidated the typographical errors and the sections which can be further optimized for gas usage below.
	
	Line 3
	`pragma abicoder v2;`
	
	`abicoder` is unnecessary as it is included automatically in the compiler after version 0.8.0.
	
	Line 21
	`liquidy`
	
	Spelling error — should be liquidity.
	
	Line 272
	`SafeERC20.safeTransferFrom(ERC20Interface(tokenAddress), from, address(this), amount)`
	
	`SafeERC20` is a library and can be used as `use SafeERC20 for ERC20Interface` which will let you write the safe transfers as `ERC20Interface(...).safeTransferFrom(...)` for readability.
	
	Line 307
	`return bytes4(keccak256("onERC721Received(address,address,uint256,bytes)"));`
	
	This can be replaced by using Solidity's built-in function `this.onERC721Received.selector;`

Line 182-188

```
function hasDoubledAllocation(uint256 tokenNumber) internal
view returns (uint256) {

    if (numberOfReallocations() % 4 < tokenNumber)
        return 0;

    return 1;
}
```

Consider returning 1 or 2 instead of 0 or 1 and remove the power calculation from the allocation size functions.

Line 81

```
10, 12, 14 and 16,
```

The comment is outdated as 16 is no longer reached.

Recommendation

Consider implementing the gas optimizations and fixing the typographical errors mentioned above.

Resolution

RESOLVED

Most of the items have been resolved.

PALADIN
BLOCKCHAIN SECURITY