



Part of Tibereum Group

# AUDITING REPORT

# Version Notes

Version	No. Pages	Date	Revised By	Notes
1.0	Total: 17	2021-06-03	Zapmore, MrTeaThyme	Final Audit
1.1	Total: 17	2021-06-29	Hebilicious, Zapmore	Added Timelock

# Audit Notes

Audit Date	2021-05-26 - 2021-05-30
Auditor/Auditors	MrTeaThyme, Hebilicious
Auditor/Auditors Contact Information	tibereum-obelisk@protonmail.com
Notes	Specified code and contracts are audited for security flaws. UI/UX (website), logic, team, and tokenomics are not audited.
Audit Report Number	OB58856821

# Disclaimer

This audit is not financial, investment, or any other kind of advice and is for informational purposes only. This report is not a substitute for doing your own research and due diligence. Obelisk is not responsible or liable for any loss, damage, or otherwise caused by reliance on this report for any purpose. Obelisk has based this audit report solely on the information provided by the audited party and on facts that existed before or during the audit being conducted. Obelisk is not responsible for any outcome, including changes done to the contract/contracts after the audit was published. This audit is fully objective and only discerns what the contract is saying without adding any opinion to it. The audit is paid by the project but neither the auditors nor Obelisk has any other connection to the project and has no obligations other than to publish an objective report. Obelisk will always publish its findings regardless of the outcome of the findings. The audit only covers the subject areas detailed in this report and unless specifically stated, nothing else has been audited. Obelisk assumes that the provided information and material were not altered, suppressed, or misleading. This report is published by Obelisk, and Obelisk has sole ownership of this report. Use of this report for any reason other than for informational purposes on the subjects reviewed in this report including the use of any part of this report is prohibited without the express written consent of Obelisk.

# Obelisk Auditing



Defi is a relatively new concept but has seen exponential growth to a point where there is a multitude of new projects created every day. In a fast-paced world like this, there will also be an enormous amount of scams. The scams have become so elaborate that it's hard for the common investor to trust a project, even though it could be legit. We saw a need for creating high-quality audits at a fast phase to keep up with the constantly expanding market. With the Obelisk stamp of approval, a legitimate project can easily grow its user base exponentially in a world where trust means everything. Obelisk Auditing consists of a group of security experts that specialize in security and structural operations, with previous work experience from among other things, PricewaterhouseCoopers. All our audits will always be conducted by at least two independent auditors for maximum security and professionalism.

As a comprehensive security firm, Obelisk provides all kinds of audits and project assistance.

# Table of Content

<b>Version Notes</b>	<b>2</b>
<b>Audit Notes</b>	<b>2</b>
<b>Disclaimer</b>	<b>2</b>
<b>Obelisk Auditing</b>	<b>3</b>
<b>Project Information</b>	<b>5</b>
<b>Executive Summary</b>	<b>6</b>
Summary Table	7
<b>Introduction</b>	<b>8</b>
<b>Findings</b>	<b>9</b>
Manual Analysis	9
Non-Descriptive Variable Names	9
Outdated Compiler Version	9
Owner Can Disable The Rewards	11
Static Analysis	12
No Findings	12
On-Chain Analysis	13
No Findings	13
<b>Appendix A - Reviewed Documents</b>	<b>14</b>
<b>Appendix B - Risk Ratings</b>	<b>15</b>
<b>Appendix C - Icons</b>	<b>15</b>
<b>Appendix D - Testing Standard</b>	<b>16</b>

# Project Information

Project Name	PaprrPrintr
Description	PaprrPrintr is an algorithmic stablecoin running on the Binance Smart Chain.
Website	<a href="https://paprrprintr.finance/">https://paprrprintr.finance/</a>
Contact	@Hash  #2935
Contact information	@Hash  #2935 on Discord
Token Name(s)	N/A
Token Short	N/A
Contract(s)	See Appendix A
Code Language	Solidity
Chain	BSC

# Executive Summary

The audit of PaprPrintr was conducted by two of Obelisks' security experts between the 26th of May 2021 and the 30th of May 2021.

**After finishing the full audit, Obelisk auditing can say that there was only a low-security issue found which poses no risk to the user's funds, in the pool contract. Low-security severity was mitigated by deploying a timelock. At the time of auditing, the timelock was set to 24h. There were no security issues found in the audited vaults.**

**The team has not reviewed the UI/UX, logic, team, or tokenomics of the PaprPrintr project.**

Please read the full document for a complete understanding of the audit.

## Summary Table

Audited Part	Severity	Note
Non-Descriptive Variable Names	Informational	N/A
Outdated Compiler Version	Informational	N/A
Owner Can Disable The Rewards	Low Risk	Mitigated

# Introduction

Obelisk was commissioned by PaprPrintr on the 25th of May 2021 to conduct a comprehensive audit of PaprPrintrs' vaults and pool on Binance Smart Chain. The following audit was conducted between the 26th of May 2021 and the 30th of May 2021 and delivered on the 3rd of June 2021. Two of Obelisk's security experts went through the related contracts using industry standards to find if any vulnerabilities could be exploited.

The comprehensive test was conducted in a specific test environment that utilized exact copies of the published contract. The auditors also conducted a manual visual inspection of the code to find security flaws that automatic tests would not find.

While conducting the audit, the Obelisk security team uses best practices to ensure that the reviewed contracts are thoroughly examined against all angles of attack. This is done by evaluating the codebase and whether it gives rise to significant risks. During the audit, Obelisk assesses the risks and assigns a risk level to each section together with an explanatory comment. Take note that the comments from the project team are their opinion and not the opinion of Obelisk.

The pool contract audited was the published contract. The vault contracts were not yet published during the audit but were checked once they are published to see that they are indeed the same as the audited contracts. The initial audit found a low severity issue in the pool that doesn't pose any risk to the user's deposited funds, more than possible opportunity cost. Low-security severity was mitigated by deploying a timelock. At the time of auditing, the timelock was set to 24h. Other informational findings are just for information and are not a security threat. We found no security issues in the audited vaults.

Please see each section of the audit to get a full understanding of the audit.



# Findings

## Manual Analysis

### Non-Descriptive Variable Names

SEVERITY	Informational
LOCATION	vaultPAPPRBUSGGETPRINR.sol Line 1014

```
1  function withdraw(uint256 _shares) public {
2      uint256 r = (balance().mul(_shares)).div(totalSupply());
3      _burn(msg.sender, _shares);
4
5      uint b = token.balanceOf(address(this));
6      if (b < r) {
7          uint _withdraw = r.sub(b);
8          IStrategy(strategy).withdraw(_withdraw);
9          uint _after = token.balanceOf(address(this));
10         uint _diff = _after.sub(b);
11         if (_diff < _withdraw) {
12             r = b.add(_diff);
13         }
14     }
15
16     token.safeTransfer(msg.sender, r);
17 }
18
```

DESCRIPTION	Variable names do not describe what they store.
RECOMMENDATION	Use descriptive variable names instead of single letters.
MITIGATED/COMMENT	N/A

## Outdated Compiler Version

SEVERITY	Informational
LOCATION	<i>Everywhere</i>

N/A

DESCRIPTION	Contracts use an older version of solidity.
RECOMMENDATION	Using up-to-date compiler versions when possible is best practice. Upgrade to Solidity 8.
MITIGATED/COMMENT	N/A

## Owner Can Disable The Rewards

SEVERITY	Mitigated ( <del>Low Risk</del> )
----------	-----------------------------------

LOCATION	<i>PaprBUSDPoolFarming.sol</i>
----------	--------------------------------

```
1  function transferAnyERC20Token(address _tokenAddr, address _to, uint _amount) public onlyOwner {
2      // require(_tokenAddr != trustedRewardTokenAddress && _tokenAddr !=
trustedDepositTokenAddress, "Cannot send out reward tokens or staking tokens!");
3
4      require(_tokenAddr != trustedDepositTokenAddress, "Admin cannot transfer out deposit tokens
from this vault!");
5      require((_tokenAddr != trustedRewardTokenAddress) || (now > adminClaimableTime), "Admin
cannot Transfer out Reward Tokens Yet!");
6      require(Token(_tokenAddr).transfer(_to, _amount), "Could not transfer out tokens!");
7  }
8
9      // function to allow owner to claim *other* modern ERC20 tokens sent to this contract
10     function transferAnyOldERC20Token(address _tokenAddr, address _to, uint _amount) public
onlyOwner {
11         // require(_tokenAddr != trustedRewardTokenAddress && _tokenAddr !=
trustedDepositTokenAddress, "Cannot send out reward tokens or staking tokens!");
12
13         require(_tokenAddr != trustedDepositTokenAddress, "Admin cannot transfer out deposit tokens
from this vault!");
14         require((_tokenAddr != trustedRewardTokenAddress) || (now > adminClaimableTime), "Admin
cannot Transfer out Reward Tokens Yet!");
15
16         OldIERC20(_tokenAddr).transfer(_to, _amount);
17     }
18 }
```

DESCRIPTION	Admin can transfer reward tokens from the vault after AdminClaimableTime passes, this poses no risk to user funds but does mean the vault can have its APY reduced to 0% given a malicious actor within the team.
-------------	---

RECOMMENDATION	AdminClaimableTime is set to 1 minute. Since the rewards token can be withdrawn to an EOA from the admin, consider implementing a timelock or a contract-based mechanism to handle these kinds of token manipulation while leaving enough time for the investors to plan ahead.
----------------	---

MITIGATED/COMMENT	Mitigated by the team on 27/06 : "Following your recommendations, I have deployed a Timelock [...] on the top of the pools [...]."
-------------------	---

# Static Analysis

No Findings

# On-Chain Analysis

No Findings




## Appendix A - Reviewed Documents

Document	Address
PAPRBU\$DpoolFarming.sol	0x5c3840f6c446E690Ba835fDa979DcceEbb08305F
StrategyPAPRBU\$DGetPRNTR.sol	0xE1fB61d67DD9148488D885c1b214b0E07ca33ca7
VaultPAPRBU\$DGetPRNTR.sol	0xCB5694c8F54Aa5f024f203e40d36696776D6F3cd
Timelock.sol	0x53eCb193D7b68Bf573aA16113D5DE6177E9C24f5

## Appendix B - Risk Ratings

Risk	Description
High Risk	A fatal vulnerability that can cause immediate loss of Tokens / Funds
Medium Risk	A vulnerability that can cause some loss of Tokens / Funds
Low Risk	A vulnerability that can be mitigated
Informational	No vulnerability

## Appendix C - Icons

Icon	Explanation
	Solved by Project Team
	Under Investigation of Project Team
	Unsolved

# Appendix D - Testing Standard

An ordinary audit is conducted using these steps.

1. Gather all information
2. Conduct a first visual inspection of documents and contracts
3. Go through all functions of the contract manually (2 independent auditors)
  - a. Discuss findings
4. Use specialized tools to find security flaws
  - a. Discuss findings
5. Follow up with project lead of findings
6. If there are flaws, and they are corrected, restart from step 2
7. Write and publish a report

During our audit, a thorough investigation has been conducted employing both automated analysis and manual inspection techniques. Our auditing method lays a particular focus on the following important concepts:

- Ensuring that the code and codebase use best practices, industry standards, and available libraries.
- Testing the contract from different angles ensuring that it works under a multitude of circumstances.
- Analyzing the contracts through databases of common security flaws.

**Follow Obelisk Auditing for the Latest Information**



ObeliskOrg



ObeliskOrg





Part of Tibereum Group