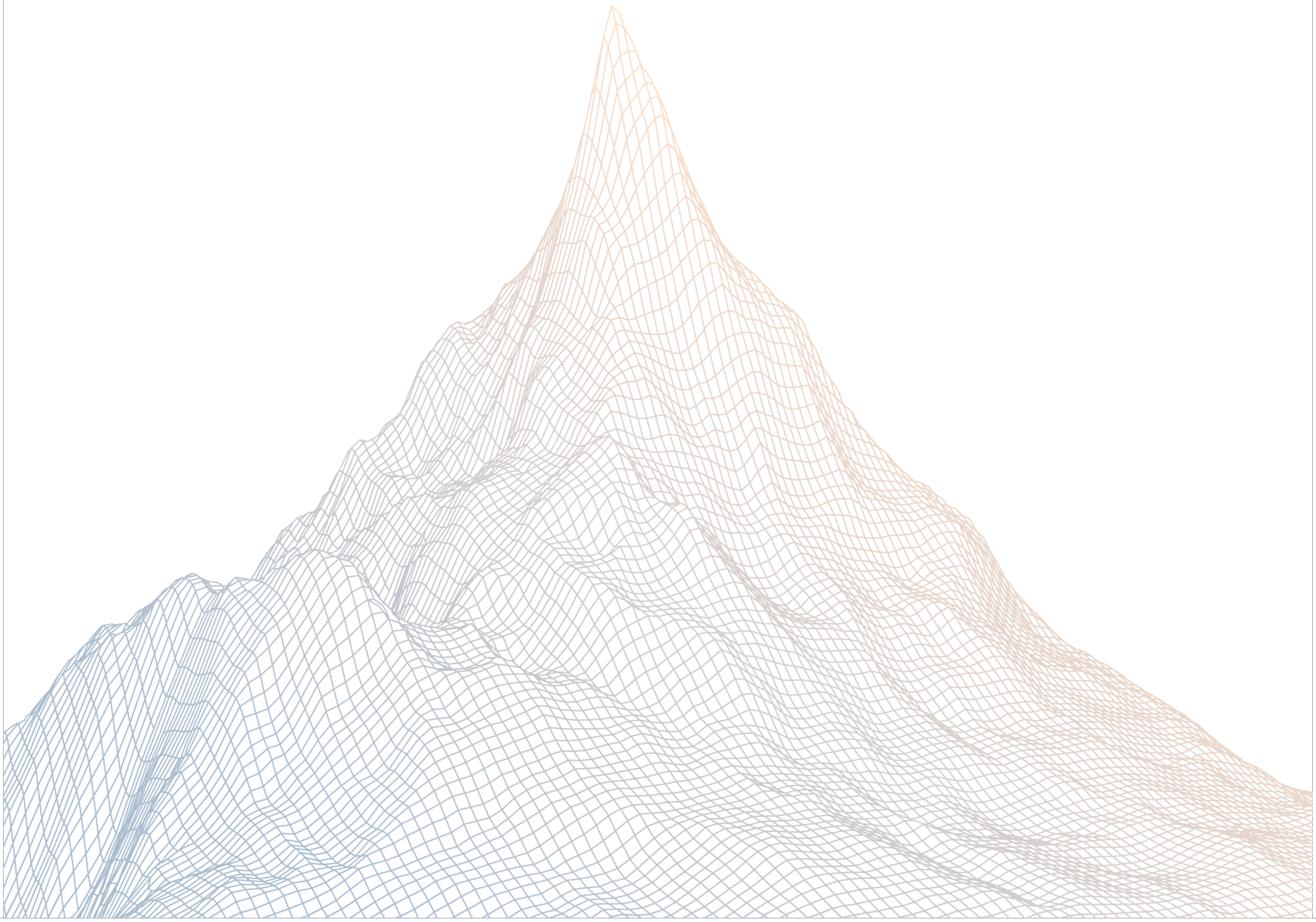
Zenith

KittyPunch

Smart Contract Security Assessment

VERSION 1.1



AUDIT DATES: February 25th to February 27th, 2025

AUDITED BY: HHK
Matte

Contents

1 Introduction

1.1 About Zenith

Zenith is an offering by Code4rena that provides consultative audits from the very best security researchers in the space. We focus on crafting a tailored security team specifically for the needs of your codebase.

Learn more about us at https://code4rena.com/zenith.

1.2 Disclaimer

This report reflects an analysis conducted within a defined scope and time frame, based on provided materials and documentation. It does not encompass all possible vulnerabilities and should not be considered exhaustive.

The review and accompanying report are presented on an "as-is" and "as-available" basis, without any express or implied warranties.

Furthermore, this report neither endorses any specific project or team nor assures the complete security of the project.

1.3 Risk Classification

SEVERITY LEVEL	IMPACT: HIGH	IMPACT: MEDIUM	IMPACT: LOW
Likelihood: High	Critical	High	Medium
Likelihood: Medium	High	Medium	Low
Likelihood: Low	Medium	Low	Low

2

Executive Summary

2.1 About KittyPunch

KittyPunch is more than just a project; it's a community-driven ecosystem that thrives on innovation and participation. Whether you are an NFT collector, a DeFi enthusiast, or a developer, there is a place for you in the KittyPunch community. Explore our documentation to learn more about how you can get involved and start earning with KittyPunch today.

2.2 Scope

The engagement involved a review of the following targets:

Target	stable-swap-factory-ng-contracts
Repository	https://github.com/Kitty-Punch/stable-swap-factory-ng-contracts
Commit Hash	`4ba3bbffd0e6021d41a54d9440fc4411fded4669`
Files	`src/*.vy`

Target	two-crypto-factory-ng-contracts
Repository	https://github.com/Kitty-Punch/two-crypto-factory-ng-contracts
Commit Hash	`743cee954188a6741c6732f4c081d928538311e9`
Files	`src/*.vy`

Target	tri-crypto-factory-ng-contracts
Repository	https://github.com/Kitty-Punch/tri-crypto-factory-ng-contracts
Commit Hash	`a7ed9c2e22e5f2ecadf8ad357063a4998eb3fe94`
Files	`src/*.vy`

Target	punch-swap-core-contracts
Repository	https://github.com/Kitty-Punch/punch-swap-core-contracts
Commit Hash	`3a8c2ed29ae400c283d219761123d953fdd258cf`
Files	`src/*.sol`

Target	punch-swap-periphery-contracts
Repository	https://github.com/Kitty-Punch/punch-swap-periphery-contracts
Commit Hash	`757a66b316a153b0bb7c6ac7b6821b2e5e26e3d2`
Files	`src/*.sol`

2.3 Audit Timeline

February 25, 2025	Audit start
February 27, 2025	Audit end
March 10, 2025	Report published

2.4 Issues Found

SEVERITY	COUNT
Critical Risk	0
High Risk	0
Medium Risk	1
Low Risk	0
Informational	1
Total Issues	**2**

3 Findings Summary

ID	Description	Status
M-1	WFLOW address for the router is incorrect	Resolved
I-1	Use env variables instead of defaulting to 0	Acknowledged

4 Findings

4.1 Medium Risk

A total of 1 medium risk findings were identified.

[M-1] WFLOW address for the router is incorrect

SEVERITY: Medium	IMPACT: Medium
STATUS: Resolved	LIKELIHOOD: Medium

Target

- KittyRouterNgPoolsOnly.vy

Description

The deployment script for the KittyRouterNgPools contract sets the `WFLOW` variable to `address(0x0)`, when the correct `WFLOW` address on the FLOW blockchain should be `0xd3bF53DAC106A0290B0483EcBC89d40FcC961f3e`.

The router contract has been deployed to mainnet at `0x87048a97526c4B66b71004927D24F61DEFcD6375` with the incorrect address value. The constructor received `address(0)` for the `WFLOW` parameter during deployment.

As a result, users cannot use the router contract's native functions to wrap or unwrap FLOW tokens. Since this variable cannot be modified after deployment, the router will remain in this broken state until a new deployment is made with the correct address.

Recommendations:

1. Update the deployment script to use the correct WFLOW address: `0xd3bF53DAC106A0290B0483EcBC89d40FcC961f3e`
2. Redeploy the router contract with the corrected parameter
3. Update any references to the old router address in other contracts or documentation
4. Consider adding validation checks in deployment scripts for critical address parameters

KittyPunch: Resolved by redeploying with correct address: 0x09d35647ceDC6725696E330Be485Ccc0D3385819

Zenith: Verified.

4.2 Informational

A total of 1 informational findings were identified.

[I-1] Use env variables instead of defaulting to 0

`SEVERITY:` Informational	`IMPACT:` Informational
`STATUS:` Acknowledged	`LIKELIHOOD:` Low

Target

- stable-swap script/*
- two-crypto script/*
- tri-crypto script/*
- punch-swap-core script/*
- punch-swap-periphery script/*

Description:

Deployment scripts in the 5 codebases all use `0x0` as default value. While it can be changed prior to deployment, it is recommended to use environment variables instead and to revert if these are not set, protecting from deploying a contract with the wrong parameters.

Recommendations:

Use `vm.envAddress(PARAM_NAME)` and revert if it's not set instead of defaulting to `0x0`.

Kittypunch: Acknowledged. We have used that approach in the past, but to be honest it was very tedious to double check each parameter in the .env every time before executing. For example, when you need to work with multiple tokens at the same time, you would need to create multiple TOKEN1_ADDRESS, TOKEN2_ADDRESS, etc.