HoneyPot Audit Report

Version 1.0

Rad

August 26, 2024

HoneyPot Audit Report

Rad

Aug 25, 2024

- Prepared by: Rad
- Lead Security Researcher: Rad

Table of Contents

Protocol Summary

This project is to enter to buy a cute honey pot NFT. The protocol should do the following:

These smart contracts are the entry points for users to participate in honeypot NFT on Ethereum, Binance Smart Chain, and Base. After participating in minting NFT, the holder can get reward by invite other users..

Disclaimer

The Rad team makes all effort to find as many vulnerabilities in the code in the given time period, but holds no responsibilities for the findings provided in this document. A security audit by the team is not an endorsement of the underlying business or product. The audit was time-boxed and the review of the code was solely on the security aspects of the Solidity implementation of the contracts.

Risk Classification

		Impact		
		High	Medium	Low
Likelihood	High	H	H/M	M
	Medium	H/M	M	M/L
	Low	M	M/L	L

We use the CodeHawks severity matrix to determine severity. See the documentation for more details.

Audit Details

Scope

- Commit Hash: 42d631c5c721c3d18c941744d8b70a94028e0959
- In Scope:

```
./src/
- HoneyPotNFT.sol
- HoneyPotPurchaseOnBase.sol
- HoneyPotPurchaseOnBNB.sol
- HoneyPotPurchaseOnETH.sol
```

Compatibilities

- Solc Version: 0.8.20
- Chain(s) to deploy contract to: Ethereum,Base and Binance Smart Chain

Roles

- Owner - Deployer of the protocol, has the power to change the owner address to the blackhole through the `renounceOwnership` function, and power change the token price, treasury address and commissionRate. and has the power to pause and unpause the contract.
- Player - Participant of the NFT Purchase, has the power to enter the contract with the `buyWithETH` function and `buyWithERC20` function.

Executive Summary

This security review journey is great. I'm glad to see that the code is well written and easy to read.

Issues found

Severity	Number of issues found
High	0
Medium	0
Low	1
Gas	0
Info	1
Total	2

Findings

Low

[L-1] Ownable parameter can’t pass the code compiler static analysis

Recommendation

It will effect the static analysis of the code.

Remove the Ownable parameter.

```
Error: Wrong argument count for modifier invocation: 1 arguments given but expected 0.
  --> src/HoneyPotPurchaseOnETH.sol:32:87:
   |
32 |     constructor(address _usdtAddress, address _usdcAddress, address _treasuryAddress) Ownable(msg.sender) {

Error: Wrong argument count for modifier invocation: 1 arguments given but expected 0.
  --> src/HoneyPotPurchaseOnBase.sol:34:85:
   |
34 |     constructor(address _usdcAddress, address _treasuryAddress, address _nftAddress) Ownable(msg.sender) {
   |


Error: Wrong argument count for modifier invocation: 1 arguments given but expected 0.
  --> src/HoneyPotPurchaseOnBNB.sol:31:87:
   |
31 |     constructor(address _usdtAddress, address _ethAddress, address _treasuryAddress) Ownable(msg.sender) {
   |
```

Information

[I-1] Using different solc version

Recommendation

The NFT contract use version `^0.8.4`, but the other contract use `0.8.20`.

[I-2] Using different import source

Recommendation

Using foundry.toml and set remappings, make the import same source or support the both import.

```
remappings = ['@openzeppelin/contracts=lib/openzeppelin-contracts/contracts']
```