AstraSec

Bearz HQ
Security Audit Report

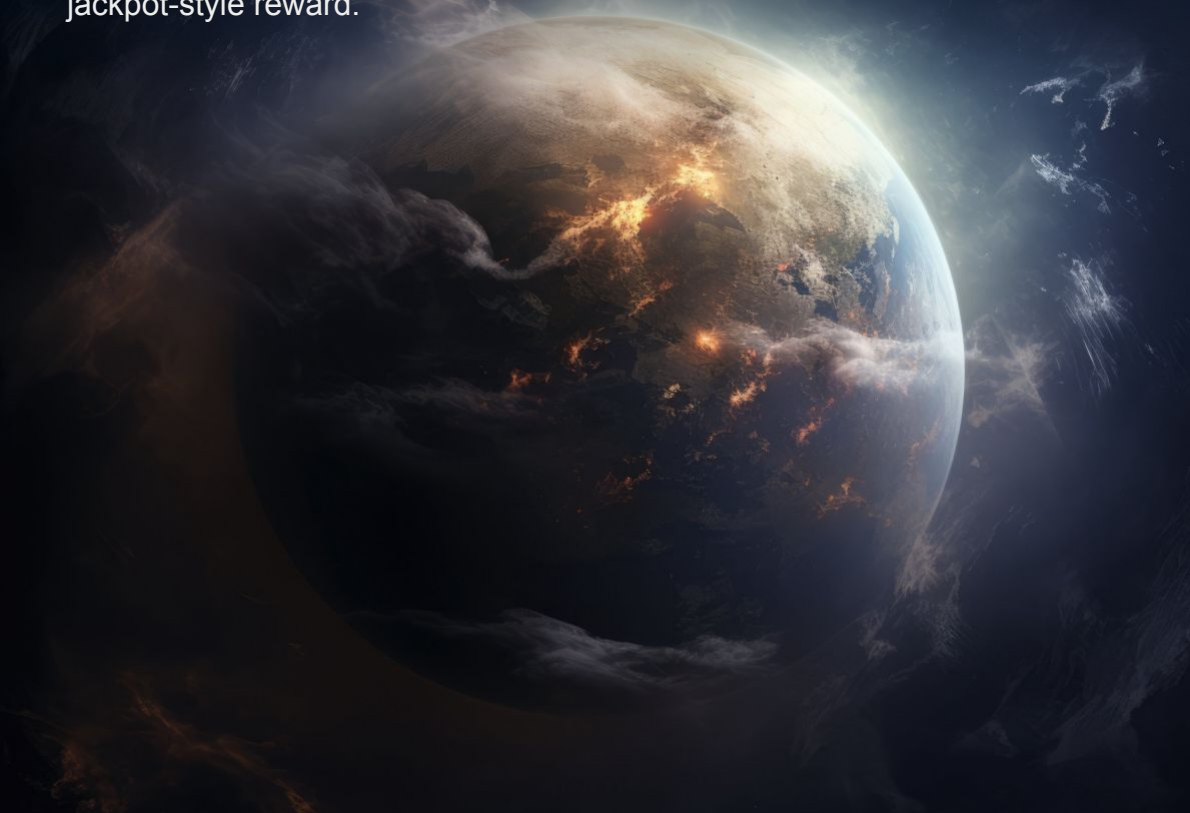
May 12, 2025

Contents

1 Introduction

1.1 About Bearz HQ

Bearz HQ is a deflationary, on-chain Player-versus-Player (PvP) survival game deployed on the Berachain blockchain. The protocol centers around Grizzly NFTs, which players mint by burning $COMB tokens and must sustain by feeding $COMB daily to prevent permanent on-chain deactivation ("death"). Players can acquire $COMB through competitive wallet raiding or by staking in the "Detox Den." The game's deflationary mechanism progressively reduces the $COMB supply through burns, culminating in an endgame where the last three wallets holding surviving Grizzly NFTs share the remaining liquidity pool, termed the "Final Fix." This strategic survival mechanic incentivizes high-stakes competition for a jackpot-style reward.



1.2 Source Code

The following source code was reviewed during the audit:

- https://github.com/chimpytuts/buzzed-beras-contracts
- CommitID: d569e44315def5ed0824514fa31328119e5e7172

And this is the final version representing all fixes implemented for the issues identified in the audit:

- https://github.com/chimpytuts/buzzed-beras-contracts
- CommitID: c74c407c3038529a8e99f4ca99aa97c49acc8adb

1.3 Revision History

Version	Date	Scope
v1.0	April 16, 2025	$COMB contract
v1.1	May 12, 2025	All smart contracts

2 Overall Assessment

This report has been compiled to identify issues and vulnerabilities within the Bearz HQ protocol. Throughout this audit, we identified a total of 5 issues spanning various severity levels. By employing auxiliary tool techniques to supplement our thorough manual code review, we have discovered the following findings.

Severity	Count	Acknowledged	Won't Do	Addressed
Critical	–	–	–	–
High	–	–	–	–
Medium	3	2	–	1
Low	2	1	–	1
Informational	–	–	–	–
Total	5	3	–	2

3 Vulnerability Summary

3.1 Overview

Click on an issue to jump to it, or scroll down to see them all.

3.2 Security Level Reference

In web3 smart contract audits, vulnerabilities are typically classified into different severity levels based on the potential impact they can have on the security and functionality of the contract. Here are the definitions for critical-severity, high-severity, medium-severity, and low-severity vulnerabilities:

Severity	Acknowledged
C-X (Critical)	A severe security flaw with immediate and significant negative consequences. It poses high risks, such as unauthorized access, financial losses, or complete disruption of functionality. Requires immediate attention and remediation.
H-X (High)	Significant security issues that can lead to substantial risks. Although not as severe as critical vulnerabilities, they can still result in unauthorized access, manipulation of contract state, or financial losses. Prompt remediation is necessary.
M-X (Medium)	Moderately impactful security weaknesses that require attention and re-mediation. They may lead to limited unauthorized access, minor financial losses, or potential disruptions to functionality.
L-X (Low)	Minor security issues with limited impact. While they may not pose significant risks, it is still recommended to address them to maintain a robust and secure smart contract.
I-X (Informational)	Warnings and things to keep in mind when operating the protocol. No immediate action required.
U-X (Undetermined)	Identified security flaw requiring further investigation. Severity and impact need to be determined. Additional assessment and analysis are necessary.

3.3 Vulnerability Details

3.3.1 [M-1] Improved Initial Liquidity Launch for $COMB

TARGET	CATEGORY	IMPACT	LIKELIHOOD	STATUS
COMB.sol	Business Logic	Medium	Low	Acknowledged

The Bearz HQ protocol utilizes $COMB, a GameFi token, which can be purchased during an ICO using $HONEY at a 1:1 price ratio via the COMB::mintForFee() function. After the ICO, 80% of the raised $HONEY and minted $COMB are allocated to establish a <$COMB, $HONEY> liquidity pool (LP) on Kodiak, with LP tokens locked by a multi-signature Protocol-Owned Liquidity (POL) wallet. A potential risk may exist in the liquidity pool initialization process. Specifically, a malicious actor could front-run the POL's liquidity addition by creating the LP on Kodiak in advance and adding minimal liquidity with an artificially high $COMB price. When the POL subsequently adds liquidity using the ICO-raised funds, the skewed price ratio would result in the POL contributing nearly all of its $HONEY but receiving disproportionately little $COMB in the pool. The attacker could then sell their $COMB at the inflated price, draining a significant portion of the $HONEY from the pool and undermining the protocol's liquidity.

buzzed-beras-contracts - COMB.sol

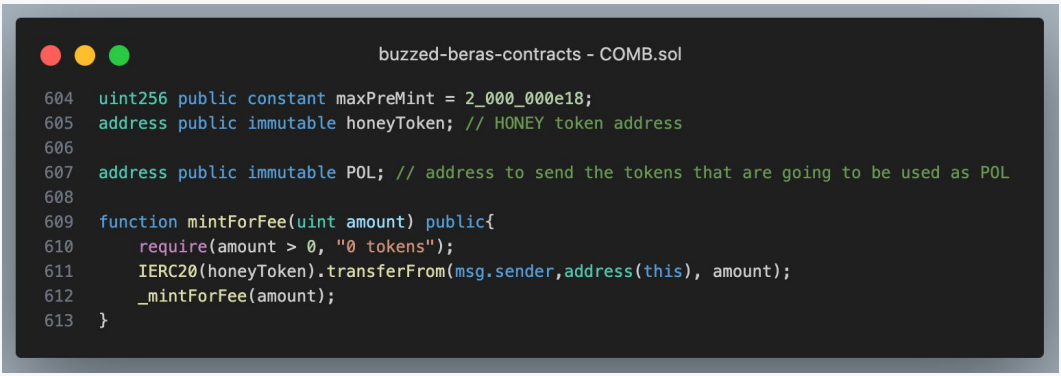
```
618  function _mintForFee(uint256 _amount) private {
619      require(finished,"You are to late");
620      require(saleStarted,"Comb nor ready yet");
621      require(preMinted + _amount <= maxPreMint,"No more comb in the market");
622      uint mintFee = _amount * mintPrice;
623      uint teamFee = mintFee * teamCut / 1000;
624
625      IERC20(honeyToken).transfer(POL, mintFee- teamFee);
626      IERC20(honeyToken).transfer(owner(), teamFee);
627
628      uint polToMint = _amount - (teamCut * _amount / 1000);
629      preMinted += _amount;
630      _mint(msg.sender, _amount);
631      _mint(POL,polToMint);
632  }
```

Remediation Implement a safeguarded liquidity launch process, check and rebalance the price before adding the initial liquidity.
Response By Team The team confirmed they will follow the safeguarded launch process.

3.3.2 [M-2] Inconsistency Between Document And Implementation

TARGET	CATEGORY	IMPACT	LIKELIHOOD	STATUS
Multiple Contracts	Coding Mistakes	Medium	Medium	Addressed

The $COMB token contract defines a constant maxPreMint variable initialized to 2,000,000, representing the maximum mint capacity for the Initial Coin Offering (ICO). However, the project's documentation states that the maximum mint capacity for the ICO is 5,000,000 $COMB. This discrepancy between the contract implementation (maxPreMint = 2,000,000) and the documentation (5,000,000 $COMB) indicates an inconsistency that could lead to operational or user trust issues.



Similar inconsistencies exist in the following cases:

- GRIZZLY Contract: The getGrizzly() function charges 100 $HONEY to mint a Grizzly NFT, which conflicts with the documented cost of 25 $HONEY.
- DETOXDEN Contract: The getCurrentRent() function calculates the daily rent for user deposits, charging 10 $HONEY per day, whereas the documentation specifies a rate of 5 $HONEY per day.

Remediation Resolve these inconsistencies by updating the contract implementations to align with the documented values.

3.3.3 [M-3] Sandwich Attack Risks in GRIZZLY and DETOXDEN

TARGET	CATEGORY	IMPACT	LIKELIHOOD	STATUS
Multiple Contracts	MEV	Medium	Medium	Acknowledged

The Bearz HQ protocol is susceptible to MEV attacks, specifically sandwich attacks, in two critical functions across the GRIZZLY and DETOXDEN contracts:

- GRIZZLY::eatComb(): Grizzly NFT owners burn $COMB tokens to feed NFTs, using tokens from other users or the liquidity pool. Burning pool $COMB raises its price relative to $HONEY, enabling MEV bots to sandwich the transaction: front-running to buy $COMB cheaply and back-running to sell at a higher price, extracting $HONEY.
- DETOXDEN::buyComb(): Swaps $HONEY in the DETOXDEN contract for $COMB via the liquidity pool to pay interest to users locking $COMB. A fixed minimum $COMB output (1e18) allows significant slippage for large $HONEY swaps, enabling MEV bots to sandwich: front-running to inflate $COMB price, reducing $COMB received, and back-running to sell at the higher price.

buzzed-beras-contracts - GRIZZLY.sol

```
2012  function _burnComb(address _victim,uint256 _combToEat) private {
2013      require(!canNotBeEaten[_victim],"can not eat from this address");
2014
2015      uint256 mintPercent = 25;
2016      bool isCombMarket = _victim == combMarket;
2017      ...
2018      ICOMB(comb).eatComb(_victim, _combToEat, msg.sender,mintPercent);
2019
2020      if(isCombMarket) {
2021          IPair(combMarket).sync();
2022      }
2023  }
```

Remediation To reduce MEV sandwich attack risks, redesign GRIZZLY::eatComb() to avoid burning $COMB from the liquidity pool. For DETOXDEN::buyComb(), replace the fixed 1e18 minimum $COMB output with a dynamic slippage threshold based on the input $HONEY amount.

Response By Team The team acknowledged that transactions can be front-run, which is normal. They plan to conduct these operations frequently to reduce slippage impact.

3.3.4 [L-1] Potential Risks Associated with Centralization

TARGET	CATEGORY	IMPACT	LIKELIHOOD	STATUS
COMB.sol	Security	Low	Low	Acknowledged

The Bearz HQ protocol exhibits significant centralization risks due to excessive administrative privileges granted to a single owner address across its core contracts, including $COMB, GRIZZLY. Enforced by the onlyOwner modifier, the owner can perform critical protocol-wide operations, such as:

- Setting the Grizzly Address: The setGrizzly() function allows the owner to update the address authorized to burn $COMB tokens.
- Toggling $COMB Consumption: The toggleCanBeEaten() function enables the owner to arbitrarily control whether $COMB held by a specific address can be consumed (e.g., burned or used in game mechanics like feeding Grizzly NFTs), potentially targeting users or manipulating game dynamics.
- Other Privileged Actions: start and stop the Initial Coin Offering (ICO), etc.

buzzed-beras-contracts - COMB.sol

```
641   /// This function is set once. It sets the comb free to be traded
642   function takeOut() public onlyOwner{
643       finished = false;
644   }
645
646   // This function allows for early deployment, then start on trigger
647   function startSale() public onlyOwner {
648       saleStarted = true;
649   }
650
651   /// This function will setup the filters for the eat the comb burn function
652   function setGrizzly(address _grizzly) public onlyOwner {
653       require(grizzly == address(0), "the farm is already built");
654       grizzly = _grizzly;
655   }
```

Remediation The centralization undermines the trustless nature expected for the Bearz HQ protocol. In order to mitigate the centralization risk, it is recommended to introduce multi-sig mechanism to undertake the role of the privileged accounts. Moreover, it is advisable to implement timelocks to govern all modifications to the privileged operations.

Response By Team This issue has been acknowledged by the team.

3.3.5 [L-2] Suggested Event Emission for Key Operations

TARGET	CATEGORY	IMPACT	LIKELIHOOD	STATUS
COMB.sol	Coding Practice	Low	Low	Addressed

The $COMB token contract, a critical component of the Bearz HQ protocol, does not emit sufficient events to log operations or state changes, particularly for privileged actions restricted by the onlyOwner modifier. We recommend emitting events for significant operations to ensure transparency, auditability, and off-chain monitoring. The absence of these events reduces the contract's observability and could hinder user trust and ecosystem integration.

buzzed-beras-contracts - COMB.sol

```
641   /// This function is set once. It sets the comb free to be traded
642   function takeOut() public onlyOwner{
643       finished = false;
644   }
645
646   // This function allows for early deployment, then start on trigger
647   function startSale() public onlyOwner {
648       saleStarted = true;
649   }
650
651   /// This function will setup the filters for the eat the comb burn function
652   function setGrizzly(address _grizzly) public onlyOwner {
653       require(grizzly == address(0), "the farm is already built");
654       grizzly = _grizzly;
655   }
```

Remediation To enhance transparency and align with smart contract best practices, we recommend implementing event emissions for all critical and privileged operations in the $COMB token contract.

4 Appendix

4.1 About AstraSec

AstraSec is a blockchain security company that serves to provide high-quality auditing services for blockchain-based protocols. With a team of blockchain specialists, AstraSec maintains a strong commitment to excellence and client satisfaction. The audit team members have extensive audit experience for various famous DeFi projects. AstraSec's comprehensive approach and deep blockchain understanding make it a trusted partner for the clients.

4.2 Disclaimer

The information provided in this audit report is for reference only and does not constitute any legal, financial, or investment advice. Any views, suggestions, or conclusions in the audit report are based on the limited information and conditions obtained during the audit process and may be subject to unknown risks and uncertainties. While we make every effort to ensure the accuracy and completeness of the audit report, we are not responsible for any errors or omissions in the report.

We recommend users to carefully consider the information in the audit report based on their own independent judgment and professional advice before making any decisions. We are not responsible for the consequences of the use of the audit report, including but not limited to any losses or damages resulting from reliance on the audit report.

This audit report is for reference only and should not be considered a substitute for legal documents or contracts.

4.3 Contact

Phone	+86 156 0639 2692
Email	contact@astrasec.ai
Twitter	https://x.com/AstraSecAI