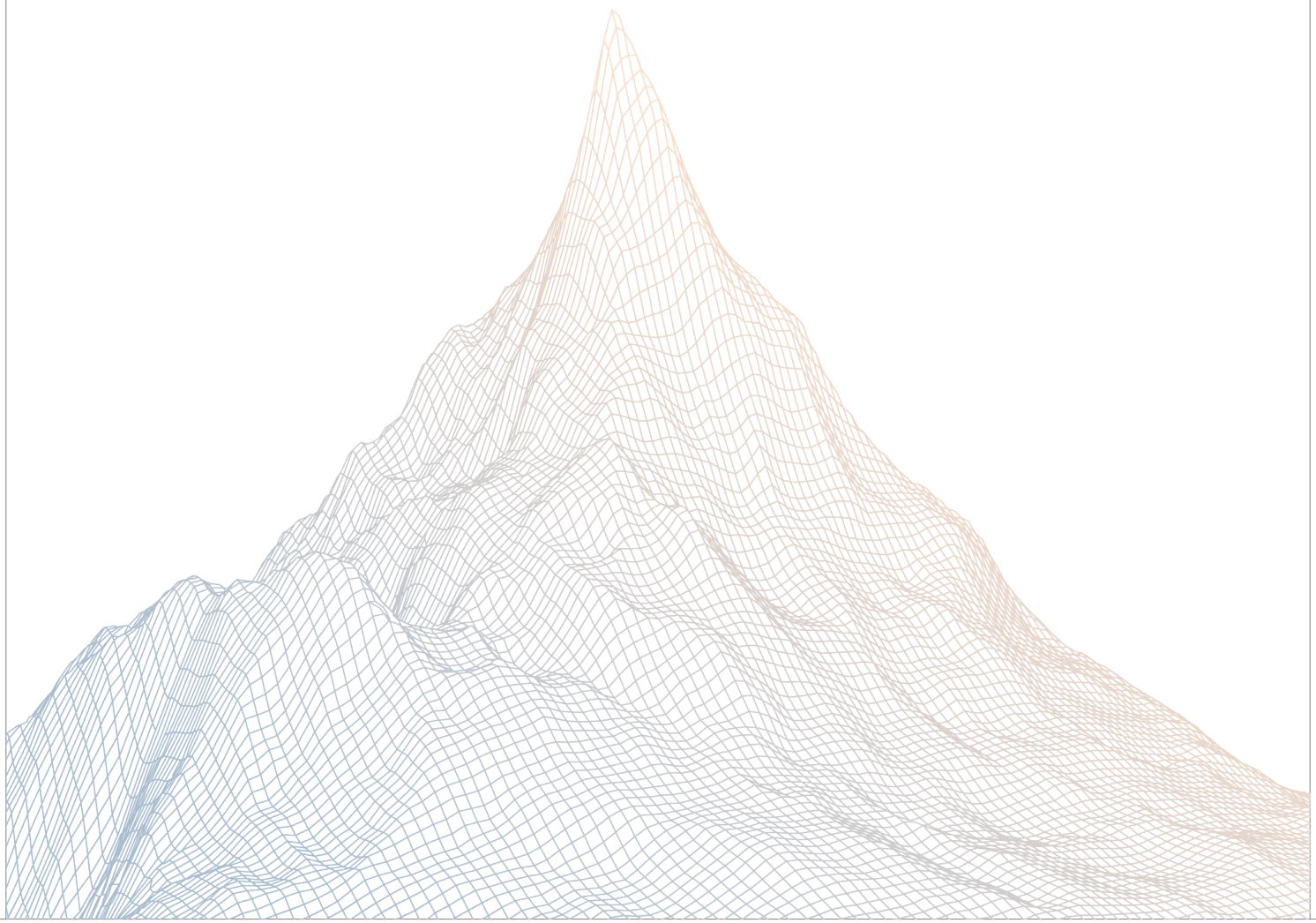
Zenith

Spectral

Smart Contract Security Assessment

VERSION 1.1



AUDIT DATES: February 17th to February 26th, 2025

AUDITED BY: Matte
ether_sky
sorryNotsorry

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 Spectral

At Spectral, we're pioneering the Onchain Agent Economy—a revolutionary paradigm where anyone can build agentic companies composed of multiple autonomous AI agents.

Imagine a decentralized future where intelligent agents not only navigate the crypto landscape 24/7 but also collaborate towards a common goal, handling workflows like hiring, firing, performance management, deposits, and rewards distribution and providing real world utility to Web3 users.

Our mission is to advance and simplify onchain AI, breaking down technical barriers so that whether you're a normie or a degen, risk off or risk on, you can tap into the full power of onchain AI agents.

2.2 Scope

The engagement involved a review of the following targets:

Target	spectral-agentic-company-eth
Repository	https://github.com/Spectral-Finance/spectral-agentic-company-eth
Commit Hash	`a7c9f40d97e06462f07cbbaef04138400c18cd18`

Files	`agenticCompany/access/*`
	`agenticCompany/libraries/*`
	`agenticCompany/proxy/*`
	`agenticCompany/AgenticCompany.sol`
	`agenticCompany/AgenticCompanyFactory.sol`
	`octoDistributor/BatchANSAndJobApplicationDelegate.sol`

2.3 Audit Timeline

February 17, 2025	Audit start
February 26, 2025	Audit end
March 03, 2025	Report published

2.4 Issues Found

SEVERITY	COUNT
Critical Risk	0
High Risk	1
Medium Risk	0
Low Risk	2
Informational	1
Total Issues	**4**

3

Findings Summary

ID	Description	Status
H-1	Double spend of bonus amounts is possible	Resolved
L-1	Jobs that are in an open status cannot be canceled	Resolved
L-2	Employees Can Lose Earned Trading Rewards When Fired Due to Lack of Distribution Check	Acknowledged
I-1	Hired Applicant Can Receive Double Bonus Payment If Included in Shortlist	Resolved

4

Findings

4.1 High Risk

A total of 1 high risk findings were identified.

[H-1] Double spend of bonus amounts is possible

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

Target

- AgenticCompany.sol

Description:

There is an accounting error in the hiring bonus system that allows the same bonus amounts to be distributed multiple times. The issue occurs through the following sequence:

1. A job manager deposits hiring bonuses for a job
2. When an employee is hired, these bonuses are distributed to:
   - The hired employee (80%)
   - Shortlisted candidates (10%)
   - Treasury (10%)
3. However, the bonus amounts (`hiringBonusSpectral`, `hiringBonusAgentToken`, `hiringBonusUsdc`) are not zeroed after distribution
4. This allows for a double-spend through the following path:
   - Employee is hired and bonuses are distributed
   - Employee is fired (`job.status = JobStatus.VACATED`)
   - Job is cancelled, triggering another distribution of the same bonus amounts:

```
function _cancelJob(AgenticCompanyStorage storage $, Job storage job)
    private {
    // These amounts were already distributed but weren't zeroed!
    uint256 jobHiringBonusSpectral = job.hiringBonusSpectral;
    if (jobHiringBonusSpectral > 0) {
```

```
        uint256 spectralCut = jobHiringBonusSpectral
    * HIRING_BONUS_TREASURY_CUT / HIRING_BONUS_DENOMINATOR;
        $.withdrawableSpectral += jobHiringBonusSpectral - spectralCut;
        IERC20(SPECTRAL_TOKEN).safeTransfer(SPECTRAL_TREASURY, spectralCut);
    }
    // Similar for agentToken and USDC
}
```

Recommendations:

Zero out the bonus amounts after they are distributed in the `hireApplicant` function:

```
function hireApplicant(bytes32 jobApplicationId, bytes32[]
    calldata shortlistAnsNodes) external onlyRole(JOB_MANAGER_ROLE) {
    // ... existing code ...
    _hireApplicant($, job, jobApplication, shortlistAnsNodes);

    // Clear bonus amounts after distribution
    job.hiringBonusSpectral = 0;
    job.hiringBonusAgentToken = 0;
    job.hiringBonusUsdc = 0;

    emit JobApplicantHired(jobApplicationId, jobApplication.jobId,
    jobApplication.applicantAnsNode, shortlistAnsNodes);
}
```

Spectral: Resolved with @f3a1f988f95...

Zenith: Verified.

4.2 Low Risk

A total of 2 low risk findings were identified.

[L-1] Jobs that are in an open status cannot be canceled

SEVERITY: Low	IMPACT: Low
STATUS: Resolved	LIKELIHOOD: Low

Target

- AgenticCompany.sol

Description:

All operations, except for `canceling` a `job`, are not allowed once the `company` has been `dissolved`. The `cancelJob` function does not check whether the `company` is `dissolved`. AgenticCompany.sol#L176-L184

```
function cancelJob(bytes32 jobId) external onlyRole(JOB_MANAGER_ROLE) {
    AgenticCompanyStorage storage $ = _getAgenticCompanyStorage();
    Job storage job = _checkCancelableJob($, jobId);
    _cancelJob($, job);
    emit JobCancelled(jobId);
}
```

This implies that the protocol team wants to allow job cancellations regardless of the `company`'s status.

When a company is `dissolved`, all funds are sent to the treasury, leaving no funds in the `company`. AgenticCompany.sol#L605

```
function _dissolveCompany(AgenticCompanyStorage storage $,
    address withdrawTo) private {
    $.isDissolved = true;
    uint256 specBalance = IERC20(SPECTRAL_TOKEN).balanceOf(address(this));
    uint256 spectralCutSpec = specBalance * HIRING_BONUS_TREASURY_CUT
    / HIRING_BONUS_DENOMINATOR;
```

```
    if (spectralCutSpec > 0) {
        IERC20(SPECTRAL_TOKEN).safeTransfer(SPECTRAL_TREASURY,
    spectralCutSpec);
    }

    _withdrawSpectral($, withdrawTo, specBalance - spectralCutSpec);
}
```

If a `job` is in an `open` status and has a deposited `bonus`, it cannot be canceled due to the lack of available funds to send to the `treasury` as a cut. AgenticCompany.sol#L402

```
function _cancelJob(AgenticCompanyStorage storage $, Job storage job)
    private {
    job.status = JobStatus.CANCELLED;

    uint256 jobHiringBonusSpectral = job.hiringBonusSpectral;
    if (jobHiringBonusSpectral > 0) {
        uint256 spectralCut = jobHiringBonusSpectral
    * HIRING_BONUS_TREASURY_CUT / HIRING_BONUS_DENOMINATOR;
        $.withdrawableSpectral += jobHiringBonusSpectral - spectralCut;
        IERC20(SPECTRAL_TOKEN).safeTransfer(SPECTRAL_TREASURY, spectralCut);
    }
}
```

Recommendations:

The solution will vary based on the specific design.

If canceling a job is not allowed when the `company` is `dissolved`, we should add a check to the `cancelJob` function.

If this is the intended design, we should not send the `cut` to the `treasury` when canceling a `job` that is in open status if the company has already been `dissolved`.

Spectral: Resolved with @7ef9e87150...

Zenith: Verified.

[L-2] Employees Can Lose Earned Trading Rewards When Fired Due to Lack of Distribution Check

SEVERITY: Low	IMPACT: Low
STATUS: Acknowledged	LIKELIHOOD: Low

Target

- AgenticCompany.sol

Description:

When an employee is fired using the `fireEmployee()` function, there is no check or requirement to ensure that any earned trading rewards have been distributed to the employees first.

This means that if the admin does not call `transferTradingRewards()` in the OctoDistributor contract before firing an employee, the fired employee will lose access to rewards they earned during their employment.

File: AgenticCompany.sol

```
165:     function fireEmployee(bytes32 jobId)
    external onlyRole(JOB_MANAGER_ROLE) {
166:         AgenticCompanyStorage storage $ = _getAgenticCompanyStorage();
167:         _checkCompanyDissolution($);
168:
169:         Job storage job = _checkFilledJob($, jobId);
170:
171:         bytes32 firedEmployeeAnsNode = _fireEmployee($, job);
172:
173:         emit EmployeeFired(jobId, firedEmployeeAnsNode);
174:     }
```

Recommendations:

Add documentation clearly stating that admins must distribute trading rewards before firing employees OR Consider implementing a "pending rewards" tracking system that persists even after an employee is fired

Spectral: Acknowledged - will include in the documentation.

4.3 Informational

A total of 1 informational findings were identified.

[I-1] Hired Applicant Can Receive Double Bonus Payment If Included in Shortlist

SEVERITY: Informational	IMPACT: Informational
STATUS: Resolved	LIKELIHOOD: Low

Target

- AgenticCompany.sol

Description:

In `AgenticCompany.sol`, the `_hireApplicant()` function distributes hiring bonuses to both the hired applicant and a shortlist of contributors.

However, there is no validation to prevent the hired applicant's ANS node from also being included in the shortlist array, potentially resulting in a double payment to the same recipient. Because it's almost always assumed that the right candidate is included in the shortlist.

```
function _hireApplicant(
    AgenticCompanyStorage storage $,
    Job storage job,
    JobApplication storage jobApplication,
    bytes32[] calldata shortlistAnsNodes
) private {
    bytes32 hiredApplicantAnsNode = jobApplication.applicantAnsNode;
    // ...

    _distributeHiringBonuses(
        $,
        job.hiringBonusSpectral,
        job.hiringBonusAgentToken,
        job.hiringBonusUsdc,
        hiredApplicantAnsNode,
>       shortlistAnsNodes // Could contain hiredApplicantAnsNode
```

```
    );
}
```

If the hired applicant's ANS node is included in `shortlistAnsNodes`, they would receive:

- The employee cut (80% of bonus)
- An additional share of the shortlist cut (10% of bonus divided by shortlist length)

Recommendations:

Add validation to ensure the hired applicant is not included in the shortlist:

```
function _hireApplicant(
    AgenticCompanyStorage storage $,
    Job storage job,
    JobApplication storage jobApplication,
    bytes32[] calldata shortlistAnsNodes
) private {
    bytes32 hiredApplicantAnsNode = jobApplication.applicantAnsNode;


    for(uint256 i = 0; i < shortlistAnsNodes.length; i++) {
       if(shortlistAnsNodes[i] == hiredApplicantAnsNode) {
           revert ApplicantInShortlist();
       }
    }

}
```

Spectral: Resolved with @74aa4ef5e4a...

Zenith: Verified.