Goldfish Audit Report

Prepared with Cyfrin

Version 2.0

Oct 10, 2025

Contents

1 About Cyfrin

Cyfrin is a Web3 security company dedicated to bringing industry-leading protection and education to our partners and their projects. Our goal is to create a safe, reliable, and transparent environment for everyone in Web3 and DeFi. Learn more about us at cyfrin.io.

2 Disclaimer

The Cyfrin team makes every effort to find as many vulnerabilities in the code as possible in the given time but holds no responsibility for the findings in this document. A security audit by the team does not endorse 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.

3 Risk Classification

	Impact: High	**Impact: Medium**	**Impact: Low**
Likelihood: High	Critical	High	Medium
Likelihood: Medium	High	Medium	Low
Likelihood: Low	Medium	Low	Low

4 Protocol Summary

The Goldfish Digital Gold Asset Ecosystem aims to create a multi-chain tokenized gold asset that ensures security, liquidity, and transparency in gold-backed digital asset management. The ecosystem will enable users to stake, redeem, and track tokenized gold assets while maintaining compliance and integrating decentralized financial (DeFi) tools.

This audit only concerned:

- `GGBR` a tokenized wrapper
- `GoldfishTokenFactory` a manager contract that allows users to mint
- `Redemption` a manage contract that allows users to redeem `GGBR`

General Overview:

Users can transfer their `GGBR` token.

Users who have satisfied KYC and other applicable regulatory requirements are able to use `Redemption flow` to:

- redeem `GGBR` to receive `Real Physical Gold`

`GGBR` function as `ERC20` tokens which users can transfer to others

Centralization Risks:

Due to the regulated nature of the underlying assets being tokenized and applicable regulatory requirements, the protocol is highly centralized by design including the ability for the protocol admins to seize the assets of users; users must place a high degree of trust in the protocol team. All issues related to centralization were outside the scope of the audit.

5 Audit Scope

The following contracts were included in the scope for this audit:

```
contracts/GodfishToken.sol
contracts/GodfishTokenFactory.sol
contracts/Redemtion.sol
contracts/KYCRegistry.sol
contracts/TransitionWallet.sol
```

6 Executive Summary

Our findings consisted of 1 Medium and 5 Low severity issues with the remainder being informational and gas optimizations.

All of the above findings were successfully mitigated by the protocol team.

Fuzz Testing:

As part of our audit we used both stateless and stateful/invariant fuzz testing; all code for our fuzz testing was delivered to the protocol team as an additional deliverable at the conclusion of the audit.

Summary

Project Name	Goldfish
Repository	Goldfish-smart-contracts
Commit	eecde79f56b37...
Audit Timeline	Oct 6th - Oct 10th
Methods	Manual Review, Cyfrin Aderyn

Issues Found

Critical Risk	0
High Risk	0
Medium Risk	1
Low Risk	5
Total Issues	6

Summary of Findings

[M-1] `Redemption.sol` have function to receive `GGBR` token but lacks a corresponding function to withdraw it, which leads to the `GGBR` being locked in the contract	Resolved
[L-1] Unsafe ERC20 Operation	Resolved
[L-2] `nonReentrant` is Not the First Modifier	Resolved
[L-3] Unused Import	Resolved
[L-4] State change without Event	Resolved
[L-5] Contract Name Reused in Different Files	Resolved

7 Findings

7.1 Medium Risk

7.1.1 Redemption.sol have function to receive GGBR token but lacks a corresponding function to withdraw it, which leads to the GGBR being locked in the contract.

Description: It appears that the contract includes a receive function to accept GGBR but lacks a corresponding function to withdraw it, which leads to the GGBR being locked in the contract. To resolve this issue, we should implement a public or external function that allows for the withdrawal of GGBR from the contract.:

Impact: Token can be locked forever in contract.

Proof of Concept: Add this drop-in PoC to Redemption.sol:

```
// Redemption.sol Line:529
function emergencyWithdraw(uint256 amount, address to) exernal onlyRole(DEFAULT_ADMIN_ROLE) {
    if (to == address(0)) revert ZeroAddress();
    if (amount == 0) InvalidAmount();

    ggbrToken.safeTransfer(to, amount);
}
```

Goldfish: Fixed in commit e6e9964

Cyfrin: Verified.

7.2 Low Risk

7.2.1 Unsafe ERC20 Operation

Description: ERC20 functions may not behave as expected. For example: return values are not always meaningful. It is recommended to use OpenZeppelin's SafeERC20 library.

Proof of Concept: Add this drop-in PoC to Redemption.sol:

```
// Redemption.sol Line:255
function _safeApprove(IERC20 token, address spender, uint256 amount) internal {
    SafeERC20.forceApprove(token, spender, 0);
    SafeERC20.forceApprove(token, spender, amount);
}
```

Goldfish: Fixed in commit 0bc73fe

Cyfrin: Verified.

7.2.2 nonReentrant is Not the First Modifier

Description: To protect against reentrancy in other modifiers, the nonReentrant modifier should be the first modifier in the list of modifiers:

```
// nonReentrant is not the first Modifier
) external whenNotPaused nonReentrant returns
```

Goldfish: Fixed in commit 458f740

Cyfrin: Verified.

7.2.3 Unused Import

Description: Redundant import statement. Consider removing it.:

```
// unused import
import "@chainlink/contracts/src/v0.8/shared/mocks/MockV3Aggregator.sol";
```

Goldfish: Fixed in commit 2837c5b

Cyfrin: Verified.

7.2.4 State Change Without Event

Description: There are state variable changes in this function but no event is emitted. Consider emitting an event to enable offchain indexers to track the changes.

Goldfish: Fixed in commit 8140c21

Cyfrin: Verified.

7.2.5 Contract Name Reused in Different Files

Description: When compiling contracts with certain development frameworks (for example: Truffle), having contracts with the same name across different files can lead to one being overwritten..

Goldfish: Fixed in commit e44c71d

Cyfrin: Verified.