WBFinance: Security Audit Report

DogScan Security Team

DogScan

Nov 4th, 2025

Contents

DogScan Security Audit Report

Project	WBFinance
Chain	BNB Smart Chain (ID 56)
Smart Contracts	`EUA` / `DonateContract` / `redemptionContract`
Audit Date	Nov 4th, 2025
Report Version	1.0
Overall Risk	**Low**

1. Executive Summary

We audited the donation and redemption contracts deployed in the WBFinance ecosystem, which rely on multiple UUPS upgradeable implementations behind ERC1967 proxies. The core product flow covers user donations, reward withdrawal, and principal redemption. This report assumes that **the project team relinquishes administrative privileges after deployment, does not add or replace redemption contracts, and refrains from extracting pooled assets**. Under this assumption, centralized governance risk is greatly reduced, while signature authorization and cumulative payout enforcement remain critical concerns.

We identified design gaps around signature domain separation, cumulative reward limits, and swap slippage protection. The team has chosen to accept these risks under current governance and operational constraints; we outline complementary hardening steps should requirements change.

2. Audit Scope

This assessment covers the following deployed contracts and their linked implementations:

- `EUA`
- `DonateContract`
  - Implementation Contract: `0xafbbd47f8c894db48297b3a308e8782e3b4ad8e1`
- `redemptionContract`
  - Implementation Contract: `0x4be29b464589f449b61c451ec56fec838480b875`

3. Methodology

We leveraged a multi-agent security analysis workflow:

1. Specialized agents first evaluated reentrancy, signature authorization, governance control, and treasury safety.
2. A lead strategist agent reviewed preliminary findings, confirmed exploitability, and prioritized impact.
3. This report summarizes confirmed issues, their context, and actionable remediation guidance.

4. Findings Overview

ID	Title	Severity	Status
L-01	Signature domain insufficient allowing cross-contract replay	**Low**	**Known & accepted by project**
L-02	Missing cumulative withdrawal cap	**Low**	**Known & accepted by project**

> Status updates reflect the project team's confirmation that admin rights are renounced and the system relies on a single redemption endpoint with off-chain controls.

> Status updates reflect the project team's confirmation that admin rights are renounced, only one redemption contract will run, and off-chain controls/cooldowns are in place.

5. Detailed Findings

L-01 Signature domain insufficient allowing cross-contract replay

Severity: Low

Description The redemption implementation relies on `DonateContract` for donation records, authorizing withdrawals with off-chain signatures. The signed payload omits the target contract address (or any domain separator) and each redemption instance maintains an isolated `_nonces`

mapping. If multiple redemption contracts are ever authorized (during staged deployments or future upgrades), a user can submit the same backend signature to each contract and withdraw repeatedly.

The team states that admin rights have been permanently renounced and only one redemption contract will ever be active. Under this operational promise the issue is currently dormant, but hardening the signature domain remains advisable in case governance needs to change in the future.

Evidence Location: `752:783:redemptionContract.sol`

```
bytes32 digest = keccak256(abi.encode(_withdrawIncome_TYPEHASH, 56,
    _specialAddress, msg.sender,orderId,incomeAmount,_nonces[msg.sender],
     deadline));
address recoveredAddress = digest.recover(signature);
...
DonateContract(donateContract).setDonateRecord(orderId,false,block.
    timestamp,incomeAmount);
_nonces[msg.sender] = _nonces[msg.sender].add(1);
```

Location: `661:670:DonateContract.sol`

```
function setRedemptionContract(address _addr, bool _isRedemptionContract)
    public onlyOwner {
    require(_addr != address(0), "Contract address cannot be zero");
    isRedemptionContract[_addr] = _isRedemptionContract;
    emit RedemptionContractUpdated(_addr, _isRedemptionContract);
}
```

Impact If governance ever adds another redemption endpoint or regains upgrade powers, signatures could be replayed across instances and drain funds.

Recommendation

- Include `address(this)` and `block.chainid` (or migrate to full EIP-712 structures) within the signed message.
- Alternatively, centralize nonce management in `DonateContract`.
- Maintain clear governance procedures if emergency redeployments become necessary.

L-02 Missing cumulative withdrawal cap

Severity: Low

Description `withdrawIncome` only ensures `incomeAmount <= ExpectedReturn` and ignores previously withdrawn rewards. If signatures are mis-issued or M-01 is exploited, users can call the function multiple times until they exceed their total expected reward.

Evidence Location: `744:749:redemptionContract.sol`

```
        require(incomeAmount<=ExpectedReturn, "Income amount cannot be greater
            than expected return");
```

Location: `885:894:DonateContract.sol`

```
    if (withdrawIncomeAmount!=0){
        donateRecords[orderId].withdrawIncomeAmount+= withdrawIncomeAmount;
    }
    donateRecords[orderId].isWithdraw = _isWithdraw;
```

Impact Without an on-chain cumulative guardrail, security hinges entirely on the accuracy of the off-chain signing service. If that service malfunctions or is compromised, the contract will still honor any excessive payout it signs.

Recommendation

- Maintain rigorous monitoring and emergency stop procedures for the signing backend.
- Consider adding a basic cumulative limit or kill switch on-chain so the system has a last-resort safeguard if off-chain controls fail.

6. Architecture and Design Review

- Contracts are structured around OpenZeppelin upgradeable modules, yielding clear separation of responsibilities.
- Backend-issued signatures remain the linchpin of security, demanding strict key management and auditing.
- Once admin permissions are renounced, ensure all critical parameters are finalized.

7. Systemic Risk Assessment

Assuming the project relinquishes administrative control:

- **Governance:** No single party can upgrade or withdraw liquidity, reducing centralized custody risk.

- **Signatures & Accounting:** Safety hinges on keeping a single redemption endpoint and well-managed signature domains.
- **Swaps & Liquidity:** The EUA cooldown helps but does not eliminate all price-manipulation scenarios.
- **Operations:** Backend signing and parameter management require tight monitoring, logging, and change control.

Overall risk is assessed as **Low** under the stated governance and operational assumptions. Regular reviews of governance commitments and off-chain controls are still recommended to prevent risk drift.

8. Recommendations

1. Explore lightweight domain binding and optional on-chain caps so governance changes can be implemented safely if needed (L-01 / L-02).
2. Harden backend signing procedures with audit trails, monitoring, and emergency stop processes.
3. Schedule regular regression reviews and expand testing around signature replay and reward limits.
4. Continue monitoring swap behaviour (including EUA cooldown effectiveness) and introduce on-chain limits or alerts if operational experience shows elevated risk.