Cyberscope

A TAC Security Company

Audit Report

Fair

October 2025

Sha256

77bcbfeff56f48aad2ee1912624715277e374aed18b513ef37efc1dbf251718e

Audited by © cyberscope

Table of Contents

Table of Contents **1**
Risk Classification **2**
Review **3**
Audit Updates 3
Source Files 3
Overview **4**
Initialization Functionality 4
Token Purchase Functionality 4
Token Redemption Functionality 5
Finalization Functionality 5
Findings Breakdown **6**
Diagnostics **7**
ZVD - Zero-Balance Vault Deallocation 8
Description 8
Recommendation 8
CCR - Contract Centralization Risk 9
Description 9
Recommendation 9
MT - Mints Tokens 10
Description 10
Recommendation 10
PAO - Potential Arbitrage Opportunities 11
Description 11
Recommendation 11
UUA - Unrevoked Update Authority 12
Description 12
Recommendation 13
Summary **14**
Disclaimer **15**
About Cyberscope **16**

Risk Classification

The criticality of findings in Cyberscope's smart contract audits is determined by evaluating multiple variables. The two primary variables are:

1. **Likelihood of Exploitation**: This considers how easily an attack can be executed, including the economic feasibility for an attacker.
2. **Impact of Exploitation**: This assesses the potential consequences of an attack, particularly in terms of the loss of funds or disruption to the contract's functionality.

Based on these variables, findings are categorized into the following severity levels:

1. **Critical**: Indicates a vulnerability that is both highly likely to be exploited and can result in significant fund loss or severe disruption. Immediate action is required to address these issues.
2. **Medium**: Refers to vulnerabilities that are either less likely to be exploited or would have a moderate impact if exploited. These issues should be addressed in due course to ensure overall contract security.
3. **Minor**: Involves vulnerabilities that are unlikely to be exploited and would have a minor impact. These findings should still be considered for resolution to maintain best practices in security.
4. **Informative**: Points out potential improvements or informational notes that do not pose an immediate risk. Addressing these can enhance the overall quality and robustness of the contract.

Severity	Likelihood / Impact of Exploitation
Critical	Highly Likely / High Impact
Medium	Less Likely / High Impact or Highly Likely/ Lower Impact
Minor / Informative	Unlikely / Low to no Impact

Review

Audit Updates

Initial Audit	05 Sep 2025 https://github.com/cyberscope-io/audits/blob/main/fair/v1/audit.pdf
Corrected Phase 2	23 Oct 2025

Source Files

Filename	SHA256
lib.rs	77bcbfeff56f48aad2ee1912624715277e374aed18b513ef37efc1dbf251718e

Overview

The **FairTokenVault** contract manages a controlled token sale and redemption system on Solana, ensuring transparent and secure handling of both **tokens** and **SOL**. It allows users to buy tokens using SOL and redeem them back on a 1:1 basis, with strict rules enforcing correct account relationships, authority management, and safe settlement. The program leverages **Program Derived Addresses (PDAs)** to hold SOL and token assets securely, using deterministic seeds for both mint authority and vault management. This design guarantees that all operations remain verifiable, consistent, and protected from unauthorized access.

Initialization Functionality

An **administrator** initializes the system by providing a pre-created SPL token mint that meets strict conditions (e.g., zero supply, correct decimals, no freeze authority). During initialization, the program transfers the mint's **MintTokens** authority from the admin to a dedicated PDA, establishes a **SOL vault PDA** and **token vault PDA**, and persists all configuration parameters in a `Config` account. This process creates a predictable and tamper-resistant foundation for token distribution, ensuring that only the program (via its PDAs) can mint or transfer tokens during the sale.

Token Purchase Functionality

The `buy_fair_token` instruction enables users to purchase tokens by sending **SOL** to the vault. Before finalization, tokens are minted directly to the user's token account under PDA authority. After finalization, purchases instead draw from the pre-minted tokens stored in the token vault. All transfers are performed through verified PDAs, guaranteeing that only the correct vaults and mints are used. The function also triggers automatic finalization once the sale period ends, locking the token supply and transitioning the system into post-sale mode.

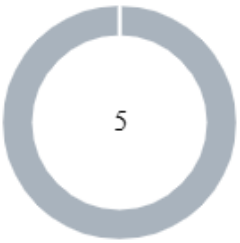
Token Redemption Functionality

The `redeem_fair_token` instruction allows users to return tokens in exchange for **SOL**. Before finalization, tokens are **burned**, permanently reducing supply; after finalization, they are transferred back into the token vault. In both cases, SOL is securely transferred from the vault PDA to the redeemer using a **System Program CPI with PDA signer seeds**, ensuring safe and authorized payments. The contract also handles the edge case where the SOL vault balance reaches zero, allowing it to be automatically re-created when new SOL is later received.

Finalization Functionality

The **finalization mechanism** locks the token's mint authority and ensures that the total supply meets a defined minimum threshold. Once triggered—either automatically after the sale period or implicitly during a buy/redeem after expiry—the contract mints any shortfall of tokens into the vault to meet the minimum supply, then revokes the mint authority entirely. After finalization, token minting becomes impossible, SOL redemptions and token transfers are routed through the vault, and the token economy enters a fully immutable state.

Findings Breakdown



Critical	0
Medium	0
Minor / Informative	5

Severity	Unresolved	Acknowledged	Resolved	Other
Critical	0	0	0	0
Medium	0	0	0	0
Minor / Informative	5	0	0	0

Diagnostics

Critical Medium Minor / Informative

Severity	Code	Description	Status
●	ZVD	Zero-Balance Vault Deallocation	Unresolved
●	CCR	Contract Centralization Risk	Unresolved
●	MT	Mints Tokens	Unresolved
●	PAO	Potential Arbitrage Opportunities	Unresolved
●	UUA	Unrevoked Update Authority	Unresolved

ZVD - Zero-Balance Vault Deallocation

Criticality	Minor / Informative
Location	lib.rs#L224
Status	Unresolved

Description

The `sol_vault` is a PDA System account with zero data. When redemptions drain it to exactly 0 lamports, the runtime will reclaim the account at the end of the transaction. That doesn't lose the address—the PDA still derives to the same pubkey—but the account object no longer exists until someone sends lamports to it again. However if any later instruction attempts to read or mutate the vault before it's re-funded, the transaction will fail because the account no longer exists on-chain.

```shell
pub fn redeem_fair_token(ctx:
Context<RedeemFairToken>, amount_to_redeem: u64)
-> Result<()>
```

Recommendation

The team should ensure to always keep a small amount of lamport to the vault to ensure that cases such as described above are avoided.

CCR - Contract Centralization Risk

Criticality	Minor / Informative
Location	lib.rs#L44
Status	Unresolved

Description

The contract's functionality and behavior are heavily dependent on external parameters or configurations. While external configuration can offer flexibility, it also poses several centralization risks that warrant attention. Centralization risks arising from the dependence on external configuration include Single Point of Control, Vulnerability to Attacks, Operational Delays, Trust Dependencies, and Decentralization Erosion.

```rust
pub fn initialize(ctx: Context<Initialize>,
sale_end: i64) -> Result<()>
```

Recommendation

To address this finding and mitigate centralization risks, it is recommended to evaluate the feasibility of migrating critical configurations and functionality into the contract's codebase itself. This approach would reduce external dependencies and enhance the contract's self-sufficiency. It is essential to carefully weigh the trade-offs between external configuration flexibility and the risks associated with centralization.

MT - Mints Tokens

Criticality	Minor / Informative
Location	lib.rs#L159
Status	Unresolved

Description

Before finalization, users are able to mint tokens by calling the `buy_fair_token` function. As a result, the contract tokens will be highly inflated.

```rust
token::mint_to(cpi_ctx, lamports_sent)?;
```

Recommendation

The team could introduce a `MAX_SUPPLY` limit to ensure that tokens minted will not be more than a reasonable amount.

PAO - Potential Arbitrage Opportunities

Criticality	Minor / Informative
Location	lib.rs#L133,224
Status	Unresolved

Description

The contract allows users to exchange SOL for the Fair Token and vice versa through the `buy_fair_token` and `redeem_fair_token` functions at a fixed 1:1 ratio. While this ensures a stable exchange rate within the contract, it does not account for potential fluctuations of the token price on external markets or DEXs. As a result, users could exploit discrepancies between the fixed rate and the market price, creating arbitrage opportunities that could be profitable at the expense of the contract's reserves.

```rust
pub fn buy_fair_token(ctx: Context<BuyFairToken>, lamports_sent: u64) -> Result<()>

pub fn redeem_fair_token(ctx: Context<RedeemFairToken>, amount_to_redeem: u64) -> Result<()>
```

Recommendation

The team could consider implementing a dynamic pricing mechanism or integrate price oracles to adjust the on-chain rate, mitigating the risk of arbitrage and protecting the contract's reserves from being drained.

UUA - Unrevoked Update Authority

Criticality	Minor / Informative
Location	lib.rs#L81
Status	Unresolved

Description

The program transfers the SPL Token `MintTokens` authority from the admin to a PDA during initialization, however it does not address the Metaplex metadata update authority associated with the provided mint. Since the mint is externally created, its metadata update authority may still belong to the original creator or another external key. This means that, even after deployment, that entity could modify the token's metadata.

```rust
let cpi_ctx = CpiContext::new(
    ctx.accounts.token_program.to_account_info(),
    SetAuthority {
        account_or_mint: mint.to_account_info(),
        current_authority:
admin.to_account_info(),
    },
);
token::set_authority(cpi_ctx,
AuthorityType::MintTokens, Some(pda))?;

mint.reload()?;
require!(
    matches!(mint.mint_authority, COption::Some(x)
if x == pda),
    ErrorCode::WrongMintAuthority
);
```

Recommendation

The team should update the contract to revoke or set the mint update authority to a PDA-controlled or zeroed-out address after initialization to ensure no further changes to the mint configuration can occur. This eliminates the risk of unauthorized minting or metadata updates and strengthens the integrity of the token distribution.

Summary

Fair Token contract implements an exchange mechanism. This audit investigates security issues, business logic concerns and potential improvements.

Disclaimer

The information provided in this report does not constitute investment, financial or trading advice and you should not treat any of the document's content as such. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes nor may copies be delivered to any other person other than the Company without Cyberscope's prior written consent. This report is not nor should be considered an "endorsement" or "disapproval" of any particular project or team. This report is not nor should be regarded as an indication of the economics or value of any "product" or "asset" created by any team or project that contracts Cyberscope to perform a security assessment. This document does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors' business, business model or legal compliance. This report should not be used in any way to make decisions around investment or involvement with any particular project. This report represents an extensive assessment process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Blockchain technology and cryptographic assets present a high level of ongoing risk Cyberscope's position is that each company and individual are responsible for their own due diligence and continuous security Cyberscope's goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies and in no way claims any guarantee of security or functionality of the technology we agree to analyze. The assessment services provided by Cyberscope are subject to dependencies and are under continuing development. You agree that your access and/or use including but not limited to any services reports and materials will be at your sole risk on an as-is where-is and as-available basis Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives false negatives and other unpredictable results. The services may access and depend upon multiple layers of third parties.

About Cyberscope

Cyberscope is a TAC blockchain cybersecurity company that was founded with the vision to make web3.0 a safer place for investors and developers. Since its launch, it has worked with thousands of projects and is estimated to have secured tens of millions of investors' funds.

Cyberscope is one of the leading smart contract audit firms in the crypto space and has built a high-profile network of clients and partners.

A **TAC Security** Company

The Cyberscope team

cyberscope.io