Cyberscope

Audit Report

Solex Launch

January 2024

SHA256 f8285ed2b4421e3efaa10270d0a0e2839437c74e243d0beb95829dfb6bbf7128

Audited by © cyberscope

Table of Contents

Table of Contents **1**
Review **2**
Audit Updates 2
Source Files 2
Overview **3**
"solex_contrib" Program Functionality 3
Findings Breakdown **4**
Diagnostics **5**
ICPC - Incomplete Claiming Process Controls 6
Description 6
Recommendation 6
PCR - Program Centralization Risk 7
Description 7
Recommendation 8
UEC - Unused Error Code 9
Description 9
Recommendation 9
USV - Unused State Variable 10
Description 10
Recommendation 10
Summary **11**
Disclaimer **12**
About Cyberscope **13**

Review

Network	SOL

Audit Updates

Initial Audit	25 Jan 2024
Corrected Phase 2	30 Jan 2024

Source Files

Filename	SHA256
programs/solex-contrib/src/lib.rs	f8285ed2b4421e3efaa10270d0a0e2839437c74e243d0beb95829dfb6bbf7128

Overview

This document presents the overview of the audit conducted for the "Solex launch" project and the "solex_contrib" program. The purpose of this audit is to identify and address security vulnerabilities, provide recommendations for code improvements, and ensure the robustness of the codebase. Recommendations have been provided to enhance security and functionality.

"solex_contrib" Program Functionality

Presale Management

The program facilitates the creation and management of presale events, enabling the setting of various parameters like total tokens for sale, soft and hard caps, and contribution limits.

Contributor Interaction

Users can participate in the presale by contributing funds, and their contributions are tracked in individual contributor profiles. The program allows for the accumulation of unclaimed tokens based on contributions.

Claiming Rewards

Contributors can claim their rewards, dependent on the claiming status having to be enabled by the presale authority.

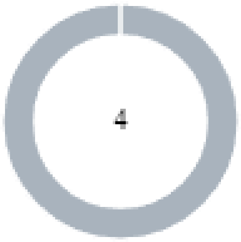
Withdrawal Functionality

The presale authority has the capability to withdraw accumulated funds from the presale event.

Administrative Controls

Functions for enabling claiming status and changing presale dates provide administrative control over the presale's lifecycle.

Findings Breakdown



●	Critical	0
●	Medium	0
●	Minor / Informative	4

	Severity	Unresolved	Acknowledged	Resolved	Other
●	Critical	0	0	0	0
●	Medium	0	0	0	0
●	Minor / Informative	4	0	0	0

Diagnostics

🔴 Critical 🟠 Medium ⚪ Minor / Informative

Severity	Code	Description	Status
⚪	ICPC	Incomplete Claiming Process Controls	Unresolved
⚪	PCR	Program Centralization Risk	Unresolved
⚪	UEC	Unused Error Code	Unresolved
⚪	USV	Unused State Variable	Unresolved

ICPC - Incomplete Claiming Process Controls

Criticality	Minor / Informative
Location	lib.rs#115
Status	Unresolved

Description

The program allows contributors to continue making contributions even after they are able to start claiming their rewards, which is achieved by the boolean variable `claiming_enabled` being set to true. This practice deviates from standard presale mechanisms where, typically, contributions are halted once the claiming phase begins. Allowing contributions to continue after the commencement of the claiming phase could lead to confusion among participants and potential discrepancies. This issue arises from the lack of a check in the `contribute` function to prevent new contributions once the claiming phase is active.

```
pub fn contribute(ctx: Context<ContributeSOL>, amount: u64) ->
Result<()> {
    ...
    Ok(())
}
```

Recommendation

To align with standard presale practices and enhance clarity for participants, it is recommended to implement additional controls within the `contribute` function. This modification will clearly separate the contribution phase from the claiming phase, preventing new contributions during the claiming period and ensuring a more structured and predictable presale process. This change will align the program with conventional presale standards.

PCR - Program Centralization Risk

Criticality	Minor / Informative
Location	lib.rs#15,84,90
Status	Unresolved

Description

The programs'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.

Furthermore, after the admin sets these key configurations, they have the authority to change some of these initial values that are crucial for the smooth functionality of the program.

Lastly, the admin has to enable the claiming of rewards that is initially disabled. Once it is enabled, it can never be disabled again.

```
pub fn create_presale(...)->Result<()>{...}

pub fn enable_claiming(ctx: Context<UpdatePresalePool>) ->
Result<()> {
    let pool = &mut ctx.accounts.presale;
    pool.claiming_enabled = true;
    Ok(())
}

pub fn change_sale_dates(
    ctx: Context<UpdatePresalePool>,
    start_timestamp: i64,
    end_timestamp: i64
) -> Result<()> {
    let pool = &mut ctx.accounts.presale;

    // Check that the start_timestamp is less than the
end_timestamp
    require!(start_timestamp < end_timestamp,
PresaleError::EndTimestampIsBeforeStartTimestamp);

    pool.start_timestamp = start_timestamp;
    pool.end_timestamp = end_timestamp;

    Ok(())
}
```

Recommendation

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

UEC - Unused Error Code

Criticality	Minor / Informative
Location	lib.rs#423
Status	Unresolved

Description

Error code `NoTokensLeft` is present that is not referenced or used anywhere in the program's logic. The presence of this unused error codes can be misleading, suggesting potential authentication checks that are not actually implemented. This could lead to a misunderstanding of the program's security features and possibly overlook actual authentication mechanisms that are in place.

```
#[msg("There are no tokens left for sale. All tokens have been
sold.")]
    NoTokensLeft
```

Recommendation

The development team should review the program to determine if this error code was intended for specific authentication checks that have not been implemented.

USV - Unused State Variable

Criticality	Minor / Informative
Location	lib.rs#229
Status	Unresolved

Description

The variable `soft_cap` within the `PresalePool` struct is defined but not effectively utilized in the program's core logic, apart from the initial validation to ensure that `soft_cap` is less than `hard_cap` . This lack of use may lead to misunderstandings about the variable's role and its impact on the program's functionality. Specifically, `soft_cap` is traditionally understood in presale contexts to represent the minimum amount of funds the presale aims to raise, and its underutilization could imply missed checks or underdeveloped features related to the presale's financial goals.

```
soft_cap: u64
```

Recommendation

The development team should reassess the intended functionality associated with `soft_cap` . If it is meant to play a significant role in the presale logic, such as affecting the distribution of tokens or influencing the continuation of the presale, appropriate logic should be implemented to reflect this. Alternatively, if `soft_cap` does not serve a purpose in the current iteration of the program, it should be removed to avoid confusion and reduce unnecessary complexity in the program's state.

Summary

Solex Launch establishes a solid foundation for managing presale events within the Solana ecosystem.

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 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.

The Cyberscope team

https://www.cyberscope.io