Cyberscope

Audit Report

TheTrumpToken

September 2024

Repository	https://github.com/KM-TrumpToken/Smart-Contracts/tree/dev/asnan
Commit	b8d8f103d0d0f5299ffaa891404537d86c779473
Audited by	© cyberscope

Table of Contents

Table of Contents 1
Risk Classification 2
Review 3
Audit Updates 3
Source Files 3
Socials 3
Overview 4
Additional Note 4
Findings Breakdown 5
Diagnostics 6
PAR - Potential Arbitrage Risk 7
Description 7
Recommendation 8
Team Update 8
PCR - Program Centralization Risk 9
Description 9
Recommendation 11
ESA - Excessive Space Allocation 12
Description 12
Recommendation 12
HTD - Hardcoded Token Decimals 13
Description 13
Recommendation 13
MC - Missing Check 14
Description 14
Recommendation 14
VAUD - Vulnerabilities and Unmaintained Dependencies 15
Description 15
Recommendation 15
Summary 16
Disclaimer 17
About Cyberscope 18

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

Repository	https://github.com/KM-TrumpToken/Smart-Contracts/tree/dev/asnan
Commit	b8d8f103d0d0f5299ffaa891404537d86c779473
Network	SOL

Audit Updates

Initial Audit	12 Sept 2024 https://github.com/cyberscope-io/audits/blob/main/great/v1/audit.pdf
Corrected Phase 2	20 Sept 2024

Source Files

Filename	SHA256
programs/trump_token/src/lib.rs	d2dacc4a4e486eae16f57ce390b443d132993bb19c80aa5cc411505d9431414e

Socials

Website	thetrumptoken.com
X (Twitter)	x.com/thetrumptoken
Telegram	t.me/thetrumptokenofficial
Discord	discord.gg/thetrumptoken

Overview

TheTrumpToken’s smart contract is designed to facilitate an Initial Coin Offering (ICO) on the Solana blockchain using the Anchor framework. The contract allows users to participate in the ICO by purchasing tokens with either SOL, USDT, or USDC. It implements functions for transferring tokens from the admin's account to a program's token account (ATA) to manage the ICO, as well as for users to deposit funds and receive tokens in return.

Key functionalities include token purchasing mechanisms for multiple assets (SOL, USDT, USDC) using real-time price feeds from the Pyth network. The contract also supports the handling of administrative operations such as the withdrawal of remaining tokens after the ICO, updating ICO-related data like the end time or token price, and managing key accounts for administrators and managers.

Additionally, the contract makes use of several token-related operations, including splitting funds between administrative and funding accounts and transferring ICO tokens to users after purchase. It incorporates security checks to ensure that only the authorized accounts can execute specific functions, such as token withdrawals and updates to ICO data. Overall, the contract is designed to facilitate an ICO process while ensuring token distribution to users based on the prevailing market prices of SOL, USDT, and USDC.

Additional Note

The configuration of the token involved in this ICO is considered out of scope. It is important to note that tokens can have a mint authority, which grants the ability to mint new tokens, potentially increasing the token supply. Additionally, there may be a freeze authority associated with the token. This authority can restrict certain wallet addresses from sending or transacting with their tokens, effectively freezing their assets. These features should be carefully reviewed and understood, as they can significantly impact the token's functionality and security.

Findings Breakdown

Critical	0
Medium	1
Minor / Informative	5

Severity	Unresolved	Acknowledged	Resolved	Other
Critical	0	0	0	0
Medium	0	1	0	0
Minor / Informative	5	0	0	0

Diagnostics

Critical Medium Minor / Informative

Severity	Code	Description	Status
Medium	PAR	Potential Arbitrage Risk	Acknowledged
Minor / Informative	PCR	Program Centralization Risk	Unresolved
Minor / Informative	ESA	Excessive Space Allocation	Unresolved
Minor / Informative	HTD	Hardcoded Token Decimals	Unresolved
Minor / Informative	MC	Missing Check	Unresolved
Minor / Informative	VAUD	Vulnerabilities and Unmaintained Dependencies	Unresolved

PAR - Potential Arbitrage Risk

Criticality	Medium
Location	lib.rs#L122,216,292
Status	Acknowledged

Description

The contract allows users to buy tokens at a price set by the admin, and these tokens are immediately transferred to the user upon purchase. While the token may not initially have a market price at the time of the ICO launch, an arbitrage risk could arise if liquidity is later added to external markets (such as decentralized exchanges) where the token could be traded. If the ICO price is lower than the token's price in these external markets, users could exploit the price discrepancy by purchasing tokens cheaply from the ICO and selling them at a higher price elsewhere.

```
pub fn buy_with_sol(
        ctx: Context<BuyWithSol>,
        _ico_ata_for_ico_program_bump: u8,
        sol_amount: u64
    ) -> ProgramResult {
    ...

 pub fn buy_with_usdt(
        ctx: Context<BuyWithUsdt>,
        _ico_ata_for_ico_program_bump: u8,
        usdt_amount: u64,
    ) -> ProgramResult {
    ...

pub fn buy_with_usdc(
    ctx: Context<BuyWithUsdc>,
    _ico_ata_for_ico_program_bump: u8,
    usdc_amount: u64,
) -> ProgramResult {
    ...
```

Recommendation

To mitigate this arbitrage risk, additional functionality should be implemented to control the immediate transfer of tokens. A potential solution is to introduce a `claim` function, allowing users to claim their tokens only after the ICO's end time. This would delay token distribution and reduce the risk of arbitrage by ensuring tokens cannot be immediately traded on external markets. The system should also keep track of each user's purchased tokens and allow them to claim these tokens only after the ICO concludes. However, introducing such a delay might also introduce a risk of insolvency if the admin withdraws tokens from the contract. To address this, safeguards should be implemented to prevent the admin from withdrawing more tokens than are needed to fulfill user claims. This will ensure that the contract remains solvent and can distribute tokens to users as promised.

Team Update

The team has acknowledged that this is not a security issue and states: *We will list tokens on different dexes after the completion of ico, so there is no chance of arbitrage for this*

PCR - Program Centralization Risk

Criticality	Minor / Informative
Location	lib.rs#L56,111,405,446,468
Status	Unresolved

Description

The program'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. Specifically, the program's functionality and behavior are heavily dependent on external parameters or configurations. These functions must be executed by a specific authorized account to set and update critical parameters within the protocol.

```
pub fn create_ico_ata(
        ctx: Context<CreateIcoATA>,
        ico_amount: u64,
        end_time:i64,
        usd_price: u64,
        funding_share: u64,
        admin: Pubkey,
        usdt_ata_for_admin: Pubkey,
        usdc_ata_for_admin: Pubkey,
        funding_account: Pubkey,
        usdt_ata_for_funding_account: Pubkey,
        usdc_ata_for_funding_account: Pubkey
    ) -> Result<()> {
    ...

pub fn update_data(ctx: Context<UpdateData>, end_time: i64, usd_price:
u64) -> Result<()> {
        // this will be used to extend time if ico event time has ended
        let data = &mut ctx.accounts.data;
        require!(
            data.admin == ctx.accounts.admin.key(),
            OwnerError::InvalidOwner
        );
        require!(
            data.end_time < Clock::get()?.unix_timestamp,
            IcoTimeError::EventNotEnded
        );



        data.end_time = end_time;
        data.usd = usd_price;

        msg!("update ico time  {}, and price {}",end_time, usd_price);
        Ok(())
    }


pub fn update_admin(ctx: Context<UpdateAdmin>,
usdt_ata_for_admin:Pubkey,new_admin: Pubkey, usdc_ata_for_admin:Pubkey)
-> ProgramResult {

        if ctx.accounts.data.admin != ctx.accounts.admin.key() {
            return Err(ProgramError::IllegalOwner);
        }
        let data = &mut ctx.accounts.data;
        data.admin = new_admin;
        data.usdt_ata_for_admin = usdt_ata_for_admin;
        data.usdc_ata_for_admin = usdc_ata_for_admin;
```

```
        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.

ESA - Excessive Space Allocation

Criticality	Minor / Informative
Location	lib.rs#L505
Status	Unresolved

Description

The program allocates more space than is necessary for the `data` account. Specifically, `data` is allocated 4500 bytes which exceeds the actual storage requirements for the data these accounts are intended to hold. Over-allocation of space leads to increased costs for account creation, as well as inefficient use of blockchain storage resources. Such practices can result in unnecessarily high transaction fees.

```
#[account(init, payer=admin, space=4500, seeds=[b"data",
admin.key().as_ref()], bump)]
pub data: Account<'info, Data>,
```

Recommendation

It is recommended to review and adjust the space allocations for both `data` account to closely match its actual data storage needs.

HTD - Hardcoded Token Decimals

Criticality	Minor / Informative
Location	lib.rs#L46,299,374
Status	Unresolved

Description

The contract currently uses hardcoded values to handle token decimals in several places, particularly when calculating token amounts. For example, the number `100000000` is used to represent the token's 8 decimal places, by being set to the constant `SCALE_FACTOR_TRUMP_TOKEN` . While this approach may work for tokens with fixed decimals, it reduces the contract's flexibility and maintainability. If the token's decimal places change in the future, this hardcoded value could lead to incorrect calculations. Relying on hardcoded values can also make the code harder to read and maintain, as it obscures the logic behind the calculations.

```
pub const SCALE_FACTOR_TRUMP_TOKEN: u64 = 100_000_000;

let ico_amount = (usdt_amount * SCALE_FACTOR_TRUMP_TOKEN)/ data.usd;

let ico_amount = (usdc_amount * SCALE_FACTOR_TRUMP_TOKEN) / data.usd;
```

Recommendation

It is recommended to replace the hardcoded decimal values with a more dynamic approach that retrieves the token's decimals directly from the token's mint account. This ensures that the correct multiplier is always applied based on the token's actual decimal places. Additionally, removing hardcoded values enhances code readability and reduces the potential for errors.

MC - Missing Check

Criticality	Minor / Informative
Location	lib.rs#L94
Status	Unresolved

Description

The contract is processing variables that have not been properly sanitized and checked that they form the proper shape. These variables may produce vulnerability issues. Specifically the `funding_share` is not checked that it doesn't exceed the maximum value of 1000.

```
data.funding_share = funding_share;
```

Recommendation

The team is advised to properly check the variables according to the required specifications.

VAUD - Vulnerabilities and Unmaintained Dependencies

Criticality	Minor / Informative
Status	Unresolved

Description

The project relies on several third-party crates that contain known security vulnerabilities or have been marked as unmaintained. These vulnerabilities could expose the project to security risks, including timing attacks, denial of service, infinite loops, and configuration corruption. Additionally, using unmaintained dependencies can introduce further risks, as they may no longer receive security patches or updates. These issues affect critical crates within the dependency tree, potentially impacting the overall security and stability of the project. Such issues can be identified by running a tool like `cargo audit`, which helps detect vulnerabilities in dependencies.

Recommendation

Review all identified vulnerabilities and unmaintained dependencies in the project's third-party crates. It is recommended to update the affected crates to the latest secure versions where possible, or consider alternative libraries that are actively maintained. Regularly monitoring and auditing third-party dependencies with tools such as `cargo audit` is crucial to ensure the project's security and integrity, reducing exposure to potential risks.

Summary

TheTrumpToken implements an ICO mechanism for the project’s token. 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 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

cyberscope.io