# CONTENTS

# Introduction

SolidProof.io is a brand of the officially registered company FutureVisions Deutschland, based in Germany. We're mainly focused on Block-chain Security such as Smart Contract Audits and KYC verification for project teams. Solidproof.io assess potential security issues in the smart contracts implementations, review for potential inconsistencies between the code base and the whitepaper/documentation, and provide suggestions for improvement.

# Disclaimer

SolidProof.io reports are not, nor should be considered, an "endorsement" or "disapproval" of any particular project or team. These reports are not, nor should be considered, an indication of the economics or value of any "product" or "asset" created by any team. SolidProof.io do not cover testing or auditing the integration with external contract or services (such as Unicrypt, Uniswap, Pancake-Swap etc'...)

**SolidProof.io Audits do not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technology proprietors. SolidProof Audits should not be used in any way to make decisions around investment or involvement with any particular project. These reports in no way provide investment advice, nor should be leveraged as investment advice of any sort.**

SolidProof.io Reports represent an extensive auditing 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. SolidProof's position is that each company and individual are responsible for their own due diligence and continuous security. SolidProof in no way claims any guarantee of security or functionality of the technology we agree to analyze.

# Project Overview

## Summary

| Project Name | Osaf Coin |
|---|---|
| Website | https://osaf.club |
| About the Project | Osaf is the living proof that being real and unique it's the key to success. Starting as a character in the "Hedz NFT Collection" created by Matt Furie, Osaf represents the excellence of the unusual. He's now ready to add value to Base community and make friends! |
| Chain | Base |
| Language | Solidity |
| Codebase | 0x72b9F9C763878964D30128f5780527392D5b3558 |
| Commit | N/A |
| Unit Tests | N/A |

## Social Medias

| | |
|---|---|
| Telegram | https://t.me/osafclub |
| Twitter | https://x.com/BasedOsaf |
| Facebook | N/A |
| Instagram | N/A |
| GitHub | N/A |
| Reddit | N/A |
| Medium | N/A |
| Discord | N/A |
| YouTube | N/A |
| TikTok | N/A |
| LinkedIn | N/A |
| CoinMarketCap | N/A |

## Audit Summary

| Version | Delivery Date | Change Log |
|---------|---------------|------------|
| v1.0 | 05 June, 2024 | • Layout Project<br><br>• Automated/ManualSecurity Review<br>• Summary |

**Note** - The following audit report presents a comprehensive security analysis of the smart contract utilized in the project. This analysis did not include functional testing (or unit testing) of the contract's logic. We cannot guarantee 100% logical correctness of the contract as it was not functionally tested by us.

## File Overview

The Team provided us with the files that should be tested in the security assessment. This audit covered the following files listed below with a SHA-1 Hash.

1. see codebase

*Please note: Files with a different hash value than in this table have been modified after the security check, either intentionally or unintentionally. A different hash value may (but need not) be an indication of a changed state or potential vulnerability that was not the subject of this scan.*

## Imported packages

Used code from other Frameworks/Smart Contracts (direct imports).

1. see codebase

*Please note: Files with a different hash value than in this table have been modified after the security check, either intentionally or unintentionally. A different hash value may (but need not) be an indication of a changed state or potential vulnerability that was not the subject of this scan.*

# Audit Information

## Vulnerability & Risk Level

Risk represents the probability that a certain source-threat will exploit vulnerability, and the impact of that event on the organization or system. Risk Level is computed based on CVSS version 3.0.

| Level | Value | Vulnerability | Risk (Required Action) |
|---|---|---|---|
| Critical | | | Immediate action to reduce risk level. |
| | 9 - 10 | | |
| High | | | Implementation of corrective actions as soon as possible. |
| | 7 - 8.9 | | |
| Medium | | | Implementation of corrective actions in a certain period. |
| | 4 - 6.9 | | |
| Low | | | Implementation of certain corrective actions or |

| | | |
|---|---|---|
| | 2 - 3.9 | accepting the risk. |
| | A vulnerability that can disrupt the contract functioning in a number of scenarios, or creates a risk that the contract may be broken. | |
| | 0 - 1.9 | An observation that does not determine a level of risk. |
| Informational | A vulnerability that affects the desired outcome when using a contract, or provides the opportunity to use a contract in an unintended way. | |
| | A vulnerability that could affect the desired outcome of executing the contract in a specific scenario. | |
| | A vulnerability that does not have a significant impact on possible scenarios for the use of the contract and is probably subjective. | |
| | A vulnerability that have informational character but is not effecting any of the code. | |

## Auditing Strategy and Techniques Applied

Throughout the review process, care was taken to check the repository for security- related issues, code quality, and compliance with specifications and best practices. To this end, our team of experienced pen-testers and smart contract developers reviewed the code line by line and documented any issues discovered. We check every file manually. We use automated tools only so that they help us achieve faster and better results.

## Methodolgy

The auditing process follows a routine series of steps:

1. Code review that includes the following:

   a. Reviewing the specifications, sources, and instructions provided to SolidProof to ensure we understand the size, scope, and functionality of the smart contract.

   b. Manual review of the code, i.e., reading the source code line by line to identify potential vulnerabilities.

   c. Comparison to the specification, i.e., verifying that the code does what is described in the specifications, sources, and instructions provided to SolidProof.

2. Testing and automated analysis that includes the following:

   a. Test coverage analysis, which determines whether test cases actually cover code and how much code is executed when those test cases are executed.

   b. Symbolic execution, which is analysing a program to determine what inputs causes each part of a program to execute.

3. Review best practices, i.e., review smart contracts to improve efficiency, effectiveness, clarity, maintainability, security, and control based on best practices, recommendations, and research from industry and academia.

4. Concrete, itemized and actionable recommendations to help you secure your smart contracts.
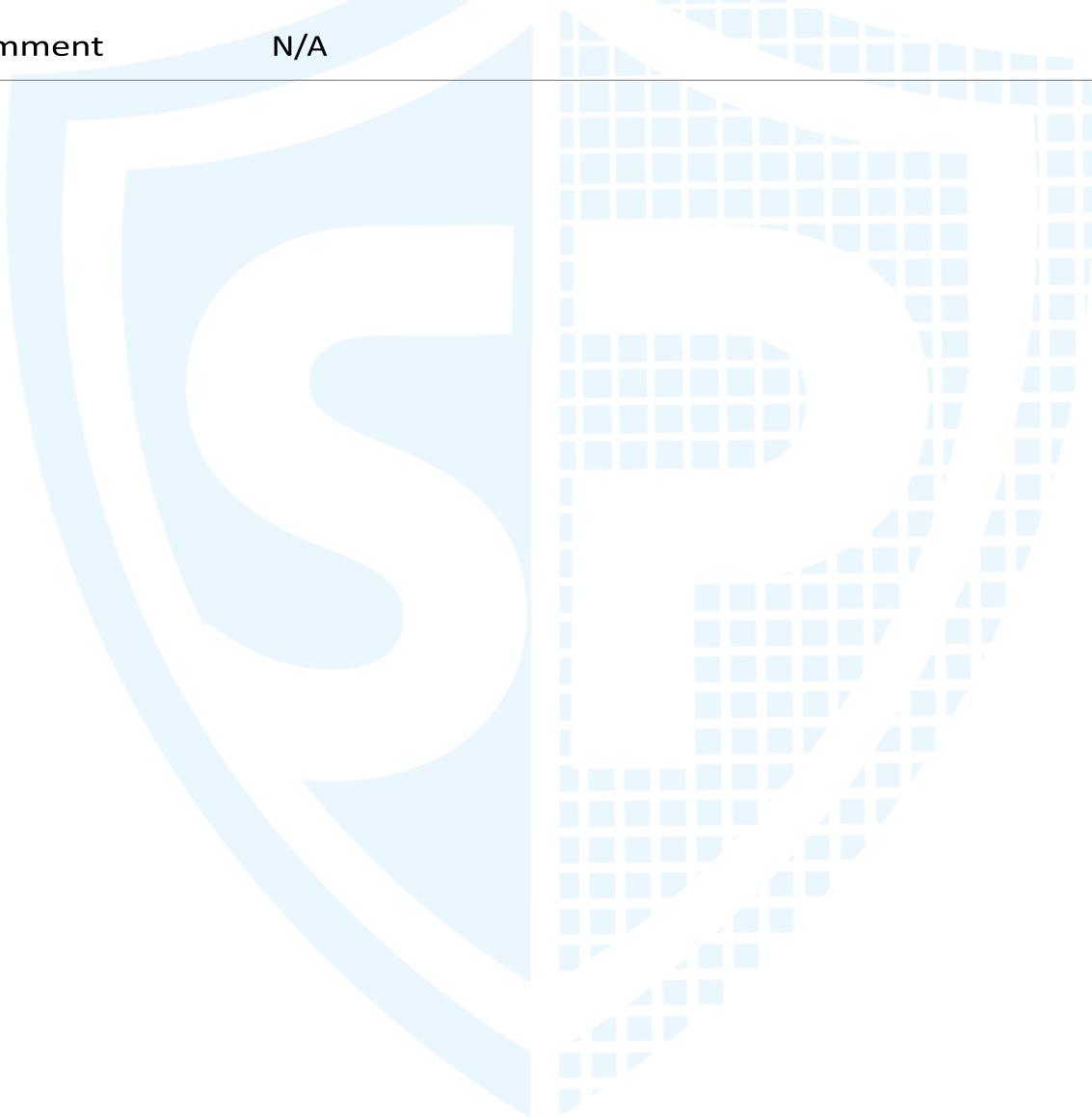
# Overall Security

## Upgradeability

| | | |
|---|---|---|
| **Contract is not upgradeable** | ✅ | **Deployer cannot add new functionalites** |

| | |
|---|---|
| Description | The contract is not an upgradeable contract. The deployer is not able to change or add any functionalities to the contract after deploying. |
| Comment | N/A |

## Ownership

| The Ownership is renounced | ✅ Ownership renounced |
|---|---|

Description   The owner renounced the ownership that means the contract's owner will no longer have any control or authority over the contract's operations.

| Comment | N/A |
|---------|-----|

**Note** - If the contract is not deployed then we would consider the owner ship to be not renounced. Moreover, if there are no ownership functionalities then the ownership is automatically considered renounced. In case of Solana SPL-Tokens a fully renounced ownership contains revoked mintAuthority, freezeAuthority or in case of SPL-Token-2022 standart the renounce of every authority which gives significant control.

## Ownership Privileges

These functions can be dangerous. Please note that abuse can lead to financial loss. We have a guide where you can learn more about these Functions.

## Minting tokens

*Minting tokens refers to the process of creating new tokens in a cryptocurrency or blockchain network. This process is typically performed by the project's owner or a designated authority, who has the ability to add new tokens to the network's total supply.*

| Contract owner cannot mint new tokens ✅ | The owner cannot mint new Tokens |
|---|---|
| Description | The owner is not able to mint new tokens once the contract is deployed. |
| Comment | mintAuthority revoked |

# Burning tokens

**Contract owner cannot burn to-The owner** ✅ **cannot burn tokens kens**

*Burning tokens is the process of permanently destroying a certain number of tokens, reducing the total supply of a cryptocurrency or token. This is usually done to increase the value of the remaining tokens, as the reduced supply can create scarcity and potentially drive up demand.*

| | |
|---|---|
| Description | The owner is not able burn tokens without any allowances. |
| Comment | N/A |

## Blacklist addresses

*Blacklisting addresses in smart contracts is the process of adding a certain address to a blacklist, effectively preventing them from accessing or participating in certain functionalities or transactions within the contract. This can be useful in preventing fraudulent or malicious activities, such as hacking attempts or money laundering.*

| **Contract Owner cannot blacklist addresses** | ✅ | **The owner cannot blacklist addresses** |
|---|---|---|
| Description | | The owner is not able blacklist addresses to lock funds. |
| Comment | | freezeAuthority revoked |

## Fees and Tax

*In some smart contracts, the owner or creator of the contract can set fees for certain actions or operations within the contract. These fees can be used to cover the cost of running the contract, such as paying for gas fees or compensating the contract's owner for their time and effort in developing and maintaining the contract.*

| Contract owner cannot set fees ✅ | The contract has 0 fee, and the owner cannot set fees |
|---|---|
| Description | The token has 0 fee. The owner cannot set fees |
| Comment | No fees or taxes implemented |

# Lock User Funds

**Contract owner cannot lock the contract**

*In a smart contract, locking refers to the process of restricting access to certain tokens or assets for a specified period of time. When tokens or assets are locked in a smart contract, they cannot be transferred or used until the lock-up period has expired or certain conditions have been met.*

| Description | The owner is not able to lock the contract by any functions or updating any variables. |
| --- | --- |
| Comment | freezeAuthority revoked |

# Centralization Privileges

*Centralization can arise when one or more parties have privileged access or control over the contract's functionality, data, or decision-making. This can occur, for example, if the contract is controlled by a single entity or if certain participants have special permissions or abilities that others do not.*

In the project there are authorities that has the authority over the following functions:

| File/Role | Privileges |
|---|---|
| Main {updateAuthority} | Change Metadata |

**Recommendations**

To avoid potential hacking risks, it is advisable for the client to manage the private key of the privileged account with care. Additionally, we recommend enhancing the security practices of centralized privileges or roles in the protocol through a decentralized mechanism or smart-contract-based accounts, such as multi-signature wallets.

Here are some suggestions what the client can do.

- Consider using multi-signature wallets: Multi-signature wallets require multiple parties to sign off on a transaction before it can be executed, providing an extra layer of security e.g. Gnosis Safe

- Use of a timelock at least with a latency of e.g. 48-72 hours for awareness on privileged operations

- Introduce a DAO/Governance/Voting module to increase transperancy and user involvement

- Consider Renouncing the ownership so that the owner cannot modify any state variables of the contract anymore. Make sure to set up everything before renouncing.

# Audit Results

## Critical issues

## High issues

| No high issues |
|:---:|

## Medium issues

| No medium issues |
|:---:|

## Low issues

| No low issues |
|:---:|

## Informational issues

### #1 | Mutable Metadata

| File | Severity | Location | Status |
|------|----------|----------|--------|
| Main | informational | – | open |

**Description**- The metadata of this token are set to mutable, which means the updateAuthority is able to change every metadata, including token-name and symbol.

| Attribute or Symbol | Meaning |
|---|---|
| **Open** | The issue is not fixed by the project team. |
| **Fixed** | The issue is fixed by the project team. |
| **Acknowledged(ACK)** | The issue has been acknowledged or declared as part of business logic. |

**Legend for the Issue Status**

**Blockchain Security | Smart Contract Audits | KYC Development | Marketing**