SECURITY ASSESSMENT

HyperSwap AI Staking

September 18, 2025

Audit Status: Pass

CFG Ninja Verified on September 18, 2025

HyperSwap AI

Executive Summary

TYPES	ECOSYSTEM	LANGUAGE
DeFi	Solana	RUST

Timeline

Audit Request	Onboarding Process	Audit Preview	Audit Release
2025-09-12	2025-09-12	2025-09-12	2025-09-18

Vulnerability Summary

7	7	0	0
Total Findings	Resolved	Pending	Unresolved

● 1 Critical	1 Resolved, 0 Pending	Critical risks are the most severe and can have a significant impact on the smart contracts functionality, security, or the entire system. These vulnerabilities can lead to the loss of user funds, unauthorized access, or complete system compromise.
● 1 High	1 Resolved, 0 Pending	High-risk vulnerabilities have the potential to cause significant harm to the smart contract or the system. While not as severe as critical risks, they can still result in financial losses, data breaches, or denial of service attacks.
● 2 Medium	1 Resolved, 0 Pending	Medium-risk vulnerabilities pose a moderate level of risk to the smart contracts security and functionality. They may not have an immediate and severe impact but can still lead to potential issues if exploited. These risks should be addressed to ensure the contracts overall security.
● 2 Low	2 Resolved, 0 Pending	Low-risk vulnerabilities have a minimal impact on the smart contracts security and functionality. They may not pose a significant threat, but it is still advisable to address them to maintain a robust security posture.

ⓘ 1 Informational	1 Resolved, 0 Pending 🟩	Informational risks are not actual vulnerabilities but provide useful information about potential improvements or best practices. These findings may include suggestions for code optimizations, documentation enhancements, or other non-critical areas for improvement.

PROJECT OVERVIEW | HyperSwap AI.

Token Summary

Parameter	Result
Address	
Name	HyperSwap AI
Token Tracker	HyperSwap AI (HyperSwap AI)
Decimals	0
Supply	0
Platform	Solana
Compiler	NONE
Contract Name	HyperSwap AI
Optimization	Yes with 200 runs
LicenseType	MIT
Language	RUST
Codebase	https://github.com/doctadg/hyper-staking

PROJECT OVERVIEW | HyperSwap AI.

Token Summary - Solana

Parameter	Result
Address	
Name	HyperSwap AI
Token Tracker	HyperSwap AI (HyperSwap AI)
Decimals	0
Supply	0
Platform	Solana
Program	
Creator Name	TBD
Creation Site	TBD
Language	RUST
Image	Not Live
Metadata File Type	JSON
Solana Source	Not Live.

PROJECT OVERVIEW | HyperSwap AI.

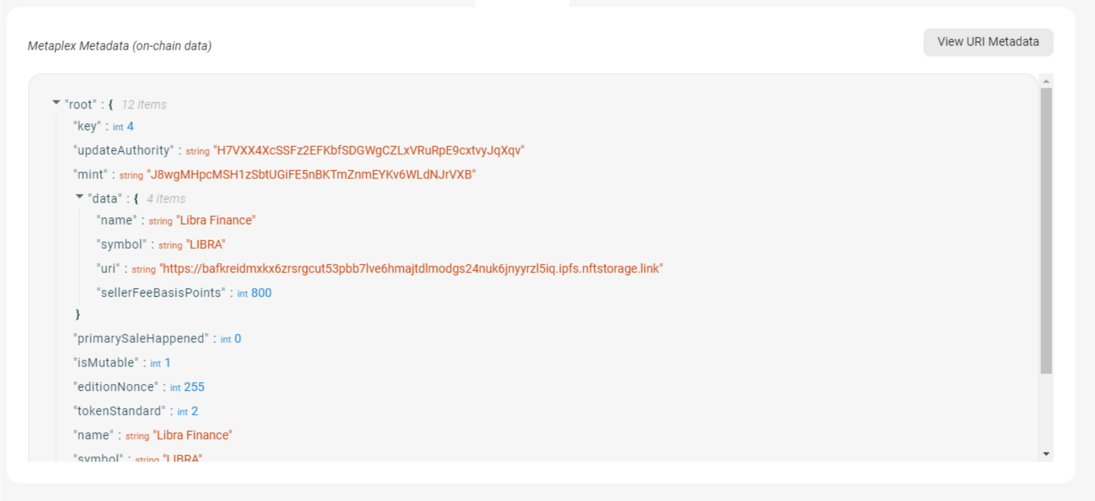
Metaplex Metadata (on-chain data)

Solana metadata refers to the additional information associated with a digital asset or NFT (Non-Fungible Token) on the Solana blockchain. It includes details such as the name, description, image, attributes, and other relevant data about the asset.

In the context of Solana, metadata is typically stored in a JSON format and is linked to the asset's unique identifier or token ID. This metadata provides important information about the asset, allowing users and applications to understand and interact with it.

Solana metadata can be used for various purposes, including displaying asset information in marketplaces, creating rich visual representations of NFTs, and enabling advanced functionalities like royalties, provenance tracking, and interoperability across different platforms.

It's worth noting that the specific structure and content of Solana metadata can vary depending on the project or application that utilizes the Solana blockchain.



Metaplex Metadata (on-chain data)

```
▾ "root" : {  12 items
    "key" : int 4
    "updateAuthority" : string "H7VXX4XcSSFz2EFKbfSDGWgCZLxVRuRpE9cxtvyJqXqv"
    "mint" : string "J8wgMHpcMSH1zSbtUGiFE5nBKTmZnmEYKv6WLdNJrVXB"
  ▾ "data" : {  4 items
        "name" : string "Libra Finance"
        "symbol" : string "LIBRA"
        "uri" : string "https://bafkreidmxkx6zrsrgcut53pbb7lve6hmajtdlmodgs24nuk6jnyyrzl5iq.ipfs.nftstorage.link"
        "sellerFeeBasisPoints" : int 800
    }
    "primarySaleHappened" : int 0
    "isMutable" : int 1
    "editionNonce" : int 255
    "tokenStandard" : int 2
    "name" : string "Libra Finance"
    "symbol" : string "LIBRA"
```

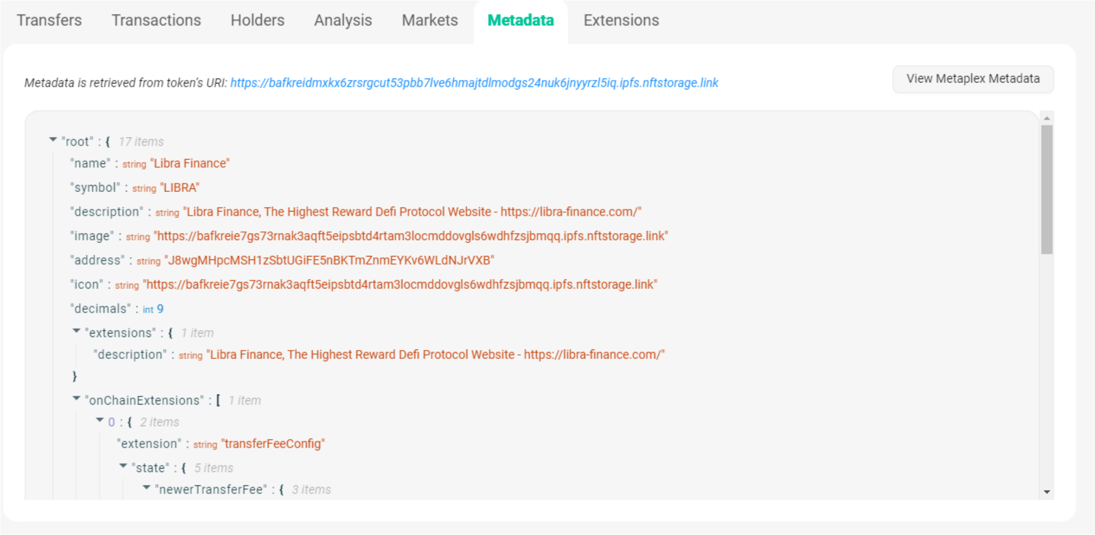
HyperSwap AI | Metadata Results.

Parameter	Value	Description
key	4	This is an integer value (int4) that represents the key associated with the root object.
updateAuthority	TBD	This is a string value that represents the update authority for the program.
mint	TBD	This is a string value that represents the mint address for the program.
name	HyperSwap AI	This is a string value that represents the name of the token.
symbol	HyperSwap AI	This is a string value that represents the symbol of the token.
uri	TBD	This is a string value that represents the URI (Uniform Resource Identifier) of the token.
sellerFeeBasisPoints	0	This is an integer value (int0) that represents the seller fee basis points for the token
primarySaleHappened	0	This is an integer value (int0) that indicates whether the primary sale of the token has happened.
isMutable	1	This is an integer value (int1) that indicates whether the token is mutable. The specific value is 1, which suggests that the token is mutable. and 0 suggest is not mutable.
editionNonce	254	This is an integer value (int255) that represents the edition nonce for the token.
tokenStandard	2	This is an integer value (int2) that represents the token standard for the program.

PROJECT OVERVIEW | HyperSwap AI.

URI Metadata

URI metadata in Solana refers to the metadata associated with a token that is retrieved from its URI (Uniform Resource Identifier). In this case, the token's URI is TBD.

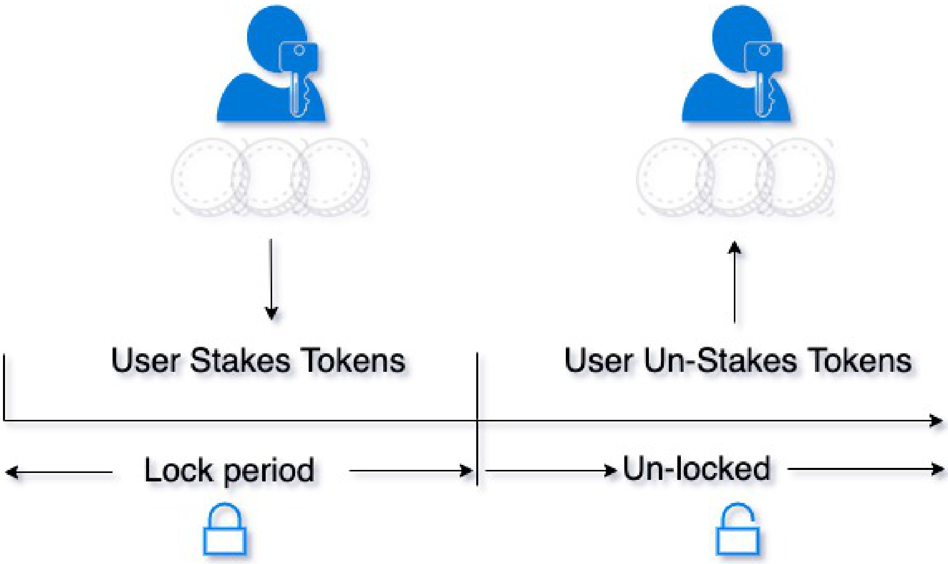


Metadata is retrieved from token's URI: https://bafkreidmxkx6zrsrgcut53pbb7lve6hmajtdlmodgs24nuk6jnyyrzl5iq.ipfs.nftstorage.link

```
▾ "root" : {  17 items
    "name" : string "Libra Finance"
    "symbol" : string "LIBRA"
    "description" : string "Libra Finance, The Highest Reward Defi Protocol Website - https://libra-finance.com/"
    "image" : string "https://bafkreie7gs73rnak3aqft5eipsbtd4rtam3locmddovgls6wdhfzsjbmqq.ipfs.nftstorage.link"
    "address" : string "J8wgMHpcMSH1zSbtUGiFE5nBKTmZnmEYKv6WLdNJrVXB"
    "icon" : string "https://bafkreie7gs73rnak3aqft5eipsbtd4rtam3locmddovgls6wdhfzsjbmqq.ipfs.nftstorage.link"
    "decimals" : int 9
  ▾ "extensions" : {  1 item
      "description" : string "Libra Finance, The Highest Reward Defi Protocol Website - https://libra-finance.com/"
  }
  ▾ "onChainExtensions" : [  1 item
    ▾ 0 : {  2 items
        "extension" : string "transferFeeConfig"
      ▾ "state" : {  5 items
        ▾ "newerTransferFee" : {  3 items
```

What is a Staking Contract?

A smart contract which allows users to stake and un-stake a specified ERC20 token. Staked tokens are locked for a specific length of time (set by the contrat owner at the outset). Once the time period has elapsed, the user can remove their tokens again.

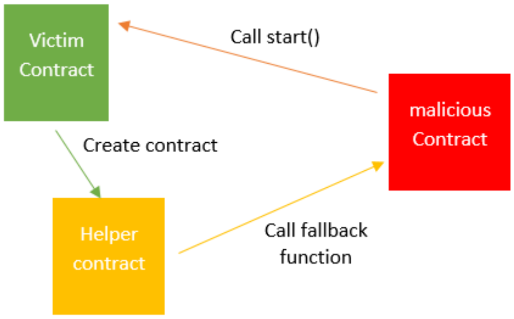


Reentrancy Check

The Project Owners of HyperSwap AI have implemented Reentrancy Guard Library

The Team has done a great job to avoid potential reentrancy issues in the contract.

You can read more about the reentrancy library used. ReentrancyGuard



TECHNICAL FINDINGS HyperSwap AI.

Smart contract security audits classify risks into several categories: Critical, High, Medium, Low, and Informational. These classifications help assess the severity and potential impact of vulnerabilities found in smart contracts.

Classification of Risk

Severity	Description
● Critical	Critical risks are the most severe and can have a significant impact on the smart contracts functionality, security, or the entire system. These vulnerabilities can lead to the loss of user funds, unauthorized access, or complete system compromise.
● High	High-risk vulnerabilities have the potential to cause significant harm to the smart contract or the system. While not as severe as critical risks, they can still result in financial losses, data breaches, or denial of service attacks.
● Medium	Medium-risk vulnerabilities pose a moderate level of risk to the smart contracts security and functionality. They may not have an immediate and severe impact but can still lead to potential issues if exploited. These risks should be addressed to ensure the contracts overall security.
● Low	Low-risk vulnerabilities have a minimal impact on the smart contracts security and functionality. They may not pose a significant threat, but it is still advisable to address them to maintain a robust security posture.
ⓘ Informational	Informational risks are not actual vulnerabilities but provide useful information about potential improvements or best practices. These findings may include suggestions for code optimizations, documentation enhancements, or other non-critical areas for improvement.

By categorizing risks into these classifications, smart contract security audits can prioritize the resolution of critical and high-risk vulnerabilities to ensure the contract's overall security and protect user funds and data.

FINDINGS

In this document, we present the findings and results of the smart contract security audit. The identified vulnerabilities, weaknesses, and potential risks are outlined, along with recommendations for mitigating these issues. It is crucial for the team to address these findings promptly to enhance the security and trustworthiness of the smart contract code.

Severity	Found	Pending	Resolved
● Critical	0	0	1
● High	0	0	1
● Medium	0	0	1
● Low	0	0	2
i Informational	0	0	1
Total	0	0	7

In a smart contract, a technical finding summary refers to a compilation of identified issues or vulnerabilities discovered during a security audit. These findings can range from coding errors and logical flaws to potential security risks. It is crucial for the project owner to thoroughly review each identified item and take necessary actions to resolve them. By carefully examining the technical finding summary, the project owner can gain insights into the weaknesses or potential threats present in the smart contract. They should prioritize addressing these issues promptly to mitigate any risks associated with the contract's security. Neglecting to address any identified item in the security audit can expose the smart contract to significant risks. Unresolved vulnerabilities can be exploited by malicious actors, potentially leading to financial losses, data breaches, or other detrimental consequences. To ensure the integrity and security of the smart contract, the project owner should engage in a comprehensive review process. This involves understanding the nature and severity of each identified item, consulting with experts if needed, and implementing appropriate fixes or enhancements. Regularly updating and maintaining the smart contract's codebase is also essential to address any emerging security concerns. By diligently reviewing and resolving all identified items in the technical finding summary, the project owner can significantly reduce the risks associated with the smart contract and enhance its overall security posture.

SOCIAL MEDIA CHECKS | HyperSwap AI.

Social Media	URL	Result
Website	https://www.hyperswap.ai/	Pass
Telegram		Pass
Twitter	https://x.com/hyperswapai	Pass
Facebook		N/A
Reddit	N/A	N/A
Instagram	N/A	N/A
CoinGecko		Fail
Github		
CMC		Fail
Email		Contact
Other		N/A

From a security assessment standpoint, inspecting a project's social media presence is essential. It enables the evaluation of the project's reputation, credibility, and trustworthiness within the community. By analyzing the content shared, engagement levels, and the response to any security-related incidents, one can assess the project's commitment to security practices and its ability to handle potential threats.

Social Media Information Notes:

Auditor Notes: Website needs a bit of improvement.

Project Owner Notes:

Assessment Results | Final Audit Score HyperSwap AI.

Review	Score
Security Score	85
Auditor Score	85

Our security assessment or audit score system for the smart contract and project follows a comprehensive evaluation process to ensure the highest level of security. The system assigns a score based on various security parameters and benchmarks, with a passing score set at 80 out of a total attainable score of 100.The assessment process includes a thorough review of the smart contracts codebase, architecture, and design principles. It examines potential vulnerabilities, such as code bugs, logical flaws, and potential attack vectors. The evaluation also considers the adherence to best practices and industry standards for secure coding. Additionally, the system assesses the projects overall security measures, including infrastructure security, data protection, and access controls. It evaluates the implementation of encryption, authentication mechanisms, and secure communication protocols. To achieve a passing score, the smart contract and project must attain a minimum of 80 points out of the total attainable score of 100. This ensures that the system has undergone a rigorous security assessment and meets the required standards for secure operation.

AUDIT
PASSED

Important Notes for HyperSwap AI

- HyperSwap AI is a staking contract deployed on the Solana blockchain, developed in Rust.
- This audit covers the main contract files, including claim_rewards.rs, deposit_rewards.rs, initialize.rs, mod.rs, pause.rs, set_purchase_time.rs, snapshot.rs, stake.rs, unstake.rs, lib.rs, and validation.rs.
- The audit was performed using tool version 3.4 on September 12, 2025.
- Updated Audit Notes (September 18, 2025):
- Most critical and high/medium risk issues have been resolved. The contract now includes proper arithmetic checks, eligibility checks, reward caps, event emission, authority validation, and access control.
- Outstanding or Partially Fixed Items:
- - No Pool Uniqueness Enforcement: PDA prevents duplicate pools, but there is no explicit check for an existing pool before initialization.
- - No Multi-Sig/Governance Support: Only a single authority is supported for admin actions. Multi-signature or DAO governance is not implemented.
- - No Historical Tracking: There are no audit trail fields for changes such as paused/unpaused times or previous authority.
- - Hardcoded Validation Constants: Validation constants are present, but not yet configurable via admin instructions.
- - No Rate Limiting/Abuse Protection: There is no rate limiting logic for sensitive instructions (e.g., claim, deposit).
- - CFG21/CFG22 (Input Validation and Balance Checks): Some improvements are present, but not all external calls and transfers have strict preconditions and validation.
- Remediation Plan:
- - Implement multi-sig/DAO governance for admin actions.
- - Add historical tracking fields and event emission for state changes.

- - Make validation constants configurable via admin instructions.
- - Add explicit pool existence checks on initialization.
- - Integrate rate limiting and abuse protection for sensitive instructions.
- - Strengthen input validation and balance checks for all external calls.
- Summary:
- The contract is much improved and addresses most major risks, but governance, historical tracking, configurability, and some validation/abuse protections remain outstanding. Remediation of these items is required for full compliance and best security practices before production deployment. Note: Audit was performed from their github repository located on https://github.com/doctadg/hyper-staking, after the contract is deployed is important to confirm code consistency.

AUDIT
PASSED

Appendix

Finding Categories

Centralization / Privilege

Centralization / Privilege findings refer to either feature logic or implementation of components that actagainst the nature of decentralization, such as explicit ownership or specialized access roles incombination with a mechanism to relocate funds.

Gas Optimization

Gas Optimization findings do not affect the functionality of the code but generate different, more optimalEVM opcodes resulting in a reduction on the total gas cost of a transaction.

Logical Issue

Logical Issue findings detail a fault in the logic of the linked code, such as an incorrect notion on howblock.timestamp works.

Control Flow

Control Flow findings concern the access control imposed on functions, such as owner-only functionsbeing invoke-able by anyone under certain circumstances.

Volatile Code

Volatile Code findings refer to segments of code that behave unexpectedly on certain edge cases that mayresult in a vulnerability.

Coding Style

Coding Style findings usually do not affect the generated byte-code but rather comment on how to makethe codebase more legible and, as a result, easily maintainable.

Inconsistency

Inconsistency findings refer to functions that should seemingly behave similarly yet contain different code,such as a constructor assignment imposing different require statements on the input variables than a setterfunction.

Coding Best Practices

ERC 20 Conding Standards are a set of rules that each developer should follow to ensure the code meet a set of creterias and is readable by all the developers.

Disclaimer

The purpose of this disclaimer is to outline the responsibilities and limitations of the security assessment and smart contract audit conducted by Bladepool/CFG NINJA. By engaging our services, the project owner acknowledges and agrees to the following terms:

1. Limitation of Liability: Bladepool/CFG NINJA shall not be held liable for any damages, losses, or expenses incurred as a result of any contract malfunctions, vulnerabilities, or exploits discovered during the security assessment and smart contract audit. The project owner assumes full responsibility for any consequences arising from the use or implementation of the audited smart contract. 2. No Guarantee of Absolute Security: While Bladepool/CFG NINJA employs industry-standard practices and methodologies to identify potential security risks, it is important to note that no security assessment or smart contract audit can provide an absolute guarantee of security. The project owner acknowledges that there may still be unknown vulnerabilities or risks that are beyond the scope of our assessment. 3. Transfer of Responsibility: By engaging our services, the project owner agrees to assume full responsibility for addressing and mitigating any identified vulnerabilities or risks discovered during the security assessment and smart contract audit. It is the project owner s sole responsibility to ensure the proper implementation of necessary security measures and to address any identified issues promptly. 4. Compliance with Applicable Laws and Regulations: The project owner acknowledges and agrees to comply with all applicable laws, regulations, and industry standards related to the use and implementation of smart contracts. Bladepool/CFG NINJA shall not be held responsible for any non-compliance by the project owner. 5. Third-Party Services: The security assessment and smart contract audit conducted by Bladepool/CFG NINJA may involve the use of third-party tools, services, or technologies. While we exercise due diligence in selecting and utilizing these resources, we cannot be held liable for any issues or damages arising from the use of such third-party services. 6. Confidentiality: Bladepool/CFG NINJA maintains strict confidentiality regarding all information and data obtained during the security assessment and smart contract audit. However, we cannot guarantee the security of data transmitted over the internet or through any other means. 7. Not a Financial Advice: Bladepool/CFG NINJA please note that the information provided in the security assessment or audit should not be considered as financial advice. It is always recommended to consult with a financial professional or do thorough research before making any investment decisions.

By engaging our services, the project owner acknowledges and accepts these terms and releases Bladepool/CFG NINJA from any liability, claims, or damages arising from the security assessment and smart contract audit. It is recommended that the project owner consult legal counsel before entering into any agreement or contract.

CFG NINJA
AUDITS