CFG NINJA

SECURITY ASSESSMENT

FalloutBase Token

May 27, 2024

Audit Status: Pass

BLADE POOL

RISK ANALYSIS | FalloutBase.

Classifications of Manual Risk Results

Classification	Description
Critical	Danger or Potential Problems.
High	Be Careful or Fail test.
Medium	Improve is needed.
Low	Pass, Not-Detected or Safe Item.
Informational	Function Detected

Manual Code Review Risk Results

Contract Security	Description
Buy Tax	0%
Sale Tax	0%
Cannot Buy	Pass
Cannot Sale	Pass
Max Tax	0%
Modify Tax	No
Fee Check	Pass
Is Honeypot?	Not Detected
Trading Cooldown	Not Detected
Enable Trade?	true
Pause Transfer?	Not Detected

Contract Security	Description
Max Tx?	Pass
Is Anti Whale?	Resolved
Is Anti Bot?	Resolved
Is Blacklist?	Resolved
Blacklist Check	Pass
is Whitelist?	Not Detected
Can Mint?	Pass
Is Proxy?	Not Detected
Can Take Ownership?	Not Detected
Hidden Owner?	Not Detected
Owner	no
Self Destruct?	Not Detected
External Call?	Not Detected
Other?	Not Detected
Holders	15
Audit Confidence	Medium
Authority Check	Pass
Freeze Check	Pass

The summary section reveals the strengths and weaknesses identified during the assessment, including any vulnerabilities or potential risks that may exist. It serves as a valuable snapshot of the overall security status of the audited project. However, it is highly recommended to read the entire security assessment report for a comprehensive understanding of the findings. The full report provides detailed insights into the assessment process, methodology, and specific recommendations for addressing the identified issues.

CFG Ninja Verified on May 27, 2024

FalloutBase

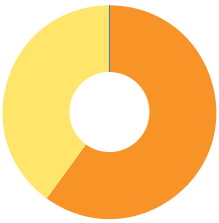
Executive Summary

TYPES	ECOSYSTEM	LANGUAGE
DeFi	BASE	Solidity

Timeline

Audit Request	Onboarding Process	Audit Preview	Audit Release
2024-04-23	2024-04-23	2024-04-23	2024-04-23

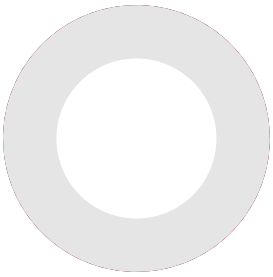
Vulnerability Summary



Total Findings	Resolved	Pending	Unresolved
	0	5	0

0 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.
0 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.
3 Medium	0 Resolved, 3 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	0 Resolved, 2 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.
0 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.

Total Unlock Progress



Unlocked	340832954273	340832.954273 %
Total Locked	420690000000	420690 %
Untracked	-761422954273	-761422.954273 %

PROJECT OVERVIEW | FalloutBase.

Token Summary

Parameter	Result
Address	0xFD025f8EdCF16871e4cc90271e44b5c7449bb121
Name	FalloutBase
Token Tracker	FalloutBase (Flo)
Decimals	18
Supply	100,000,000
Platform	BASE
Compiler	v0.8.17+commit.8df45f5f
Contract Name	FalloutBase
Optimization	Yes with 200 runs
LicenseType	MIT
Language	Solidity
Codebase	https://basescan.org/address/0xfd025f8edcf16871e4cc90271e44b5c7449bb121#code

Main Contract Assessed

Name	Contract	Live
FalloutBase	0xFD025f8EdCF16871e4cc90271e44b5c7449bb121	Yes

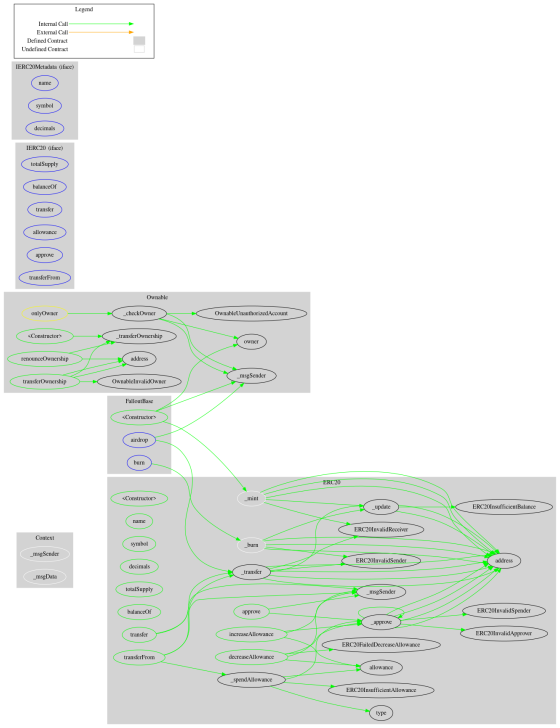
TestNet Contract Was Not Assessed

Solidity Code Provided

SolID	File Sha-1	FileName
FALLOUT	ba587a8e9bcd99a78863910be355aa13c83ff999	falloutbase.sol

Call Graph

The Smart Contract Graph is a visual representation of the interconnectedness and relationships between smart contracts within a blockchain network. It provides a comprehensive view of the interactions and dependencies between different smart contracts, allowing developers and users to analyze and understand the flow of data and transactions within the network. The Smart Contract Graph enables better transparency, security, and efficiency in decentralized applications by facilitating the identification of potential vulnerabilities, optimizing contract execution, and enhancing overall network performance.



TECHNICAL FINDINGS | FalloutBase.

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.

Flo-05 | Missing Event Emission.

Category	Severity	Location	Status
Volatile Code	Low	falloutbase.sol: L: 734 C: 14	Detected

Description

Detected missing events for critical arithmetic parameters. There are functions that have no event emitted, so it is difficult to track off-chain changes.The linked code does not create an event for the transfer.

Recommendation

Emit an event for critical parameter changes. It is recommended emitting events for the sensitive functions that are controlled by centralization roles.

Mitigation

References:

Understanding Events in Smart Contracts

Flo-14 | Unnecessary Use Of SafeMath.

Category	Severity	Location	Status
Logical Issue	Medium	falloutbase.sol: L: 38 C: 14	Acknowledge

Description

The SafeMath library is used unnecessarily. With Solidity compiler versions 0.8.0 or newer, arithmetic operations
will automatically revert in case of integer overflow or underflow.
library SafeMath {
An implementation of SafeMath library is found.
using SafeMath for uint256;
SafeMath library is used for uint256 type in contract.

Recommendation

We advise removing the usage of SafeMath library and using the built-in arithmetic operations provided by the
Solidity programming language.

Mitigation

References:

Writing Clean Code for Solidity: Best Practices for Solidity Development

Flo-19 | Centralization Privileges of Flo.

Category	Severity	Location	Status
Coding Style	Medium	falloutbase.sol: L: 734 C: 14	Detected

Description

In a smart contract, the concept of "onlyOwner" functions refers to certain functions that can only be executed by the owner or creator of the contract. These functions are typically designed to perform critical actions or modify sensitive data within the contract. By restricting access to these functions, the contract owner maintains control and ensures the integrity and security of the contract.

Function Name	Parameters	Visibility
Airdrop	airdrop	external

Recommendation

Inheriting from Ownable and calling its constructor on yours ensures that the address deploying your contract is registered as the owner. The onlyOwner modifier makes a function revert if not called by the address registered as the owner. It is important that deployr or owner secure the credentials that has owner priviledge to ensure the security of the project.

Mitigation

References:

Guide to Ownership and Access Control in Solidity

Writing Clean Code for Solidity: Best Practices for Solidity Development

Flo-20 | Unchecked Token Transfers.

Category	Severity	Location	Status
Coding Style	Low	falloutbase.sol: L: 734 C:14	Detected

Description

The airdrop function does not check the return value of the _transfer call or handle potential exceptions..

Recommendation

Implement checks to ensure token transfers are successful..

Mitigation

References:

Writing Clean Code for Solidity: Best Practices for Solidity Development

Flo-21 | Potential Underflow in Decrease Allowance .

Category	Severity	Location	Status
Coding Style	Medium	falloutbase.sol: L: 734.	Detected

Description

The decreaseAllowance function could underflow if requestedDecrease is larger than the current allowance..

Recommendation

The existing check prevents underflow but consider adding a revert message for clarity...

Mitigation

References:

Writing Clean Code for Solidity: Best Practices for Solidity Development

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	0
High	0	0	0
Medium	3	3	0
Low	2	2	0
Informational	0	0	0
Total	5	5	0

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 | FalloutBase.

Social Media	URL	Result
Website	https://www.falloutbase.com/	Pass
Telegram	https://t.me/FalloutBase	Pass
Twitter	https://twitter.com/FalloutBasee	Pass
Facebook		N/A
Reddit	N/A	N/A
Instagram	N/A	N/A
CoinGecko	N/A	N/A
Github		N/A
CMC	N/A	N/A
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 | FalloutBase.

Score Rsesults

Review	Score
Overall Score	80/100
Auditor Score	80/100

Review by Section	Score
Manual Scan Score	19
SWC Scan Score	37
Advance Check Score	24

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.

Important Notes for Flo

- Conclusion: The FalloutBase smart contract exhibits a solid foundation, leveraging OpenZeppelin's reputable ERC20 and Ownable implementations. However, several issues need addressing, such as potential denial of service in the airdrop function, gas inefficiencies, and lack of input validation. No tax mechanism is present, and the contract allows for direct, full-value transfers. The custom errors and modern Solidity practices are commendable, but improvements are necessary to enhance security and functionality. With proper fixes, the contract can be more robust and reliable for deployment.

Auditor Score =80
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.