Reflect Audit

@reflectaudit

SMART CONTRACT

Code Review and Security Analysis Report

February, 26, 2024

Token Name	REFLECT AUDIT
Network	ETH
Type	ERC20

0xd62baa6f3c579316b2798262a219b367c549c7dc

Request your audit at reflectaudit.com

Audited by © ReflectAudit

Project Overview

Summary

Project Name	REFLECT AUDIT
Website	https://www.reflectaudit.com
About	Reflect Audit is a blockchain security company dedicated to ensuring the integrity and safety of blockchain projects through comprehensive audit services.
Chain	ETH
Address	0xD62BaA6f3C579316b2798262A219B367c549C7DC

Reflect Audit, a blockchain security company, has conducted a comprehensive and multifaceted audit of the REFLECT AUDIT smart contract. The audit applied a combination of manual and automated methods to scrutinize the contract for security issues, code integrity, and adherence to industry standards and best practices.

Details

Name	Reflect Audit
Symbol	REF
Decimals	10
Supply	100,000,000
License Type	MIT
Language	Solidity
Codebase	https://etherscan.io/address/0xD62BaA6f3C579316b2798262A219B367c549C7DC#code

Social Medias

Telegram	https://t.me/ReflectAudit
Twitter	https://twitter.com/ReflectAudit
Github	https://github.com/ReflectAudit
Medium	https://medium.com/@reflectaudit
Youtube	https://www.youtube.com/@reflectaudit1

Audit Information

Risk is defined as the likelihood of a specific threat source taking advantage of a vulnerability and the consequent effect of this exploitation on the organization or system.

Audit Approach and Employed Methodologies

Throughout the assessment period, meticulous attention was dedicated to scrutinizing the repository for issues pertaining to security, code integrity, and adherence to defined standards and industry best practices. Our seasoned penetration testers and blockchain developers conducted an exhaustive, line-by-line code examination, recording any irregularities identified.

Manual inspections were carried out on each file, complemented by the use of automated tools designed to enhance the speed and effectiveness of our findings.

Audit Methodology

The audit adheres to a systematic protocol comprising:

1. **Detailed Code Analysis,** involving:

   a. Examination of provided specifications, sources, and instructions to ensure comprehensive understanding of the contract's dimensions, boundaries, and functionality.

   b. Rigorous manual scrutiny of the source code to pinpoint potential security weaknesses.

   c. Correlation with provided specifications to confirm the code's actions align with the intended functionalities.

2. **Automated Testing and Analysis**, featuring:

   a. Evaluation of test coverage to ascertain the extent to which test cases encompass the code and the quantum of code executed during these tests.

   b. Symbolic execution to analyze which inputs trigger the execution of specific segments of the program.

3. **Best Practice Review**, which includes:

   a. Examination of smart contracts to bolster efficiency, efficacy, lucidity, maintainability, and control, drawing from the best practice frameworks, insights, and scholarly work from the tech sector and academic circles.

4. **Comprehensive and Actionable Guidance** to enhance the security posture of your smart contracts.

Classifications of Manual Risk Results

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

Manual Code Review

Contract Privilege	Description
Buy Tax	20%
Sale Tax	25%
Cannot Sale	Pass
Max Tax	25%
Modify Tax	Detected
Fee Check	Pass
Is Honeypot?	Not Detected
Trading Cooldown	Not Detected
Can Pause Trade?	Not Detected
Pause Transfer?	Not Detected
Max Tx?	Detected
Is Anti Whale?	Detected
Is Anti Bot?	Not Detected
Is Blacklist?	Not Detected
Blacklist Check	Pass
is Whitelist?	Not Detected
Is Proxy?	Not Detected
Can Take Ownership?	Not detected
Hidden Owner?	Not detected
Self Destruct?	Not Detected
External Call?	Not detected
Holders	1

Smart Contract Vulnerability Checks

ID	Name	Severity
SWC-100	Function Default Visibility	Pass
SWC-101	Integer Overflow and Underflow	Pass
SWC-102	Outdated Compiler Version file	Pass
SWC-103	A floating pragma is set	Pass
SWC-104	Unchecked Call Return Value	Pass
SWC-105	Unprotected Ether Withdrawal	Pass
SWC-106	Unprotected SELFDESTRUCT Instruction	Pass
SWC-107	Read of persistent state following external call	Pass
SWC-108	State variable visibility is not set	Pass
SWC-109	Uninitialized Storage Pointer	Pass
SWC-110	Assert Violation	Pass
SWC-111	Use of Deprecated Solidity Functions	Pass
SWC-112	Delegate Call to Untrusted Callee	Pass
SWC-113	Multiple calls are executed in the same transaction	Pass
SWC-114	Transaction Order Dependence	Pass
SWC-115	Authorization through tx.origin	Pass
SWC-116	Control flow decision based on block.timestamp	Pass
SWC-117	Signature Malleability	Pass
SWC-118	Incorrect Constructor Name	Pass
SWC-119	Shadowing State Variables	Pass
SWC-120	Potential use of block.number as source of randomness	Pass
SWC-121	Missing Protection against Signature Replay Attacks	Pass
SWC-122	Lack of Proper Signature Verification	Pass
SWC-123	Requirement Violation	Pass
SWC-124	Write to Arbitrary Storage Location	Pass
SWC-125	Incorrect Inheritance Order	Pass
SWC-126	Insufficient Gas Griefing	Pass
SWC-127	Arbitrary Jump with Function Type Variable	Pass
SWC-128	DoS With Block Gas Limit	Pass
SWC-129	Typographical Error	Pass
SWC-130	Right-To-Left-Override control character (U+202E)	Pass
SWC-131	Presence of unused variables	Pass
SWC-132	Unexpected Ether balance	Pass
SWC-133	Hash Collisions with Multiple Variable Length Arguments	Pass
SWC-134	Message call with hardcoded gas amount	Pass
SWC-135	Code With No Effects (Irrelevant/Dead Code)	Pass
SWC-136	Unencrypted Private Data On-Chain	Pass

Security Findings Overview

Smart Contract Vulnerability Checks: The REFLECT AUDIT smart contract has passed checks against a comprehensive list of known vulnerabilities as per the SWC Registry, including protection against reentrancy, overflow/underflow, and invalidated inputs.

Contract Privileges: The contract includes a set of privileges reserved for the owner, such as setting a maximum tax rate of 25%, adding liquidity, withdrawing stuck ETH, and starting trading. However, it prohibits the owner from imposing unfair taxes (above 25%) and from locking the contract to restrict user access to funds.

Overall Security

Upgradeability

Note - Upgradeability refers to the ability to update or modify a contract's code after it has been deployed on the blockchain. This feature is crucial for fixing bugs, improving functionality, or adding new features to the contract over time.

Contract is not upgradeable	Deployer cannot update the contract with new functionalities
Description	The contract is not an upgradeable contract. The deployer is not able to change or add any functionalities to the contract after deploying

Ownership

Note - In the event that the contract has not been deployed, ownership is presumed to be not renounced. Additionally, if the contract lacks ownership functions, it is by default deemed to have renounced ownership.

The ownership is not renounced	The owner is not renounce
Description	The owner has not renounced the ownership that means that the owner retains control over the contract’s operations, including the ability to execute functions that may impact the contract’s users or stakeholders.

Ownership Privileges

Minting tokens

Note - In the event that the contract has not been deployed, ownership is presumed to be not renounced. Additionally, if the contract lacks ownership functions, it is by default deemed to have renounced ownership.

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

Burning tokens

Note - Minting tokens is the act of generating new tokens within a cryptocurrency or blockchain ecosystem. This activity is usually carried out by the project's owner or a specified authority, endowed with the power to augment the total supply of tokens on the network.

Contract owner cannot burn tokens	The owner cannot burn tokens
Description	The owner is not able burn tokens without any allowances.

Blacklist addresses

Note - Blacklisting addresses in smart contracts involves placing specific addresses on a blacklist, thereby restricting their ability to access or engage in certain functions or transactions within the contract. This mechanism serves as a preventive measure against fraudulent or malicious activities, including hacking attempts and money laundering, by blocking the implicated addresses from participating in the contract's operations.

Contract owner cannot blacklist addresses	The owner cannot blacklist addresses
Description	The owner is not able blacklist addresses to lock funds

Fees and Tax

Note - In certain smart contracts, the owner or creator has the authority to impose fees on specific actions or operations conducted within the contract. These fees are designed to offset operational costs associated with the contract, such as gas fees, or to remunerate the owner for the investment of time and effort in the development and ongoing maintenance of the contract.

Contract owner cannot set fees more than 25%	The owner cannot levy unfair taxes
Description	The owner is not able to set the fees above 25%

Lock Funds

Note - In the context of a smart contract, locking denotes the act of temporarily restricting the ability to access certain tokens or assets. During the lock-up period, these tokens or assets cannot be moved or utilized until either the specified duration concludes or predefined conditions are fulfilled. This mechanism is employed to ensure compliance with certain operational protocols or to secure the assets for a determined period.

Owner cannot lock the contract	The owner cannot lock the contract
Description	The owner is not able to lock the contract by any functions or updating any variables

Security Best Practices Compliance:

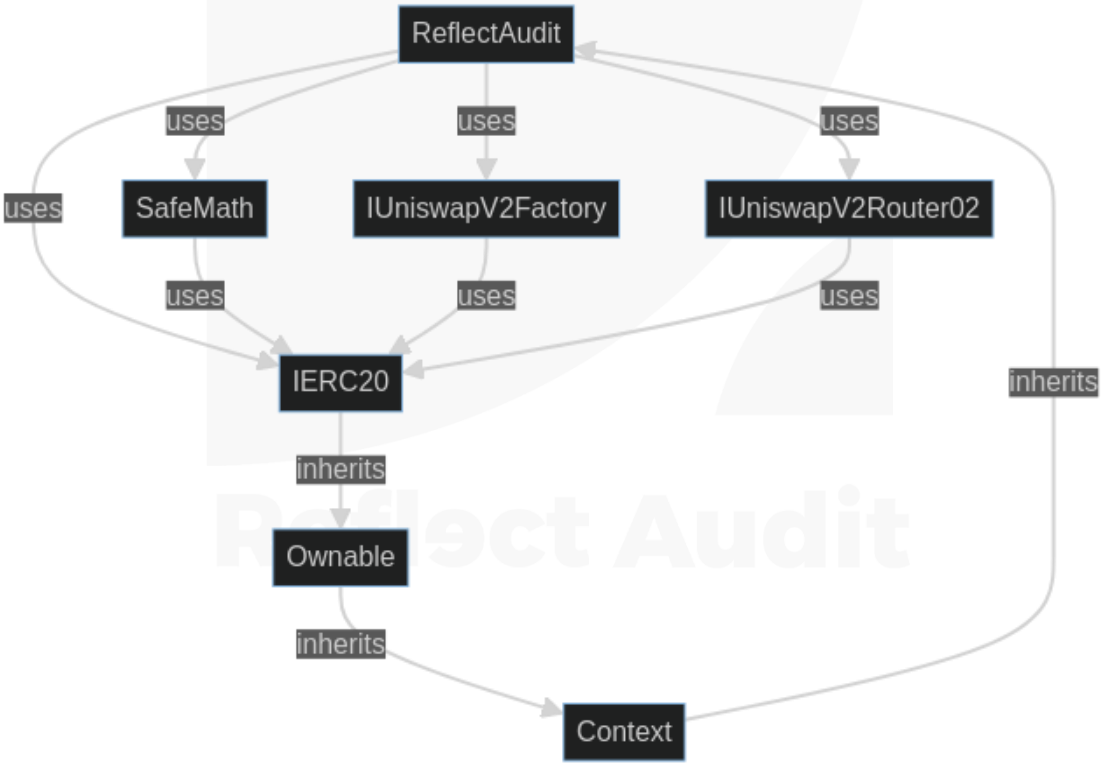
Upgradeability: The smart contract is not upgradeable, meaning that once deployed, its code cannot be altered to add new functionalities or fix potential vulnerabilities.

Ownership and Privileges: Ownership is not renounced, granting the owner significant control over the contract's operations. The owner cannot mint new tokens, burn tokens without allowances, or blacklist addresses, which may contribute to a trustful and stable environment but also limits the flexibility to address unforeseen issues.

Fees and Taxation: The contract enforces limitations on the owner's ability to set fees, capping them at a maximum of 25% to prevent exploitative taxation.

Inheritance

An inheritance graph is a visual depiction of the inheritance relationships among contracts, particularly in the context of programming languages like Solidity. In object-oriented programming, inheritance enables one class (or contract in Solidity) to acquire the attributes and behaviors (properties and methods) of another class. This graph effectively illustrates the hierarchical structure of contracts, indicating which contracts are derived from others and how they are interconnected through the inheritance mechanism. It serves as a crucial tool for understanding the architecture of complex systems where multiple contracts are interlinked, showcasing the flow of functionalities and properties from parent contracts to child contracts.



.

Centralization Privileges

Centralization within the context of a smart contract or blockchain project refers to a scenario where control or privileged access over the contract's functions, data, or decision-making processes is concentrated in the hands of one or a few parties. This condition manifests when the contract's design or governance structure allows a single entity or a select group of participants to exert significant influence or control. Examples of centralization include scenarios where only specific authorities can execute crucial contract functions, modify parameters, or access sensitive information, thereby differentiating them from other participants who lack such capabilities. This centralized control mechanism contrasts with the decentralized ethos typically associated with blockchain technology, where the aim is to distribute control and decision-making across a wider network of participants.

Privileges
• Add Liquidity to the contract before trading is open • Withdraw stuck ETH from the contract • Set Tax up to 5% • Remove Limits • Start Trading but cannot disable it

Recommendations and Conclusions

Reflect Audit recommends the following measures to ensure ongoing security and compliance:

Continuous Security Practices: Engage in regular re-audits, maintain a bug bounty program, and update security protocols as necessary to adapt to the evolving landscape of blockchain technology.

Risk Acknowledgment: Clients and users must recognize the inherent risks associated with smart contracts and blockchain technologies, including the potential for coding errors and vulnerabilities.

Final Thoughts

The REFLECT AUDIT smart contract has demonstrated adherence to security best practices and has passed extensive security checks. However, it is essential to acknowledge the immutable and non-upgradeable nature of the contract, which necessitates meticulous initial scrutiny and long-term security planning. The limitations placed on the owner's privileges could be seen as a commitment to ensuring a stable and trustful environment for users. It is advised that potential investors conduct their due diligence and consider the audit findings as part of a comprehensive investment decision-making process.

Reflect Audit Disclaimer

Please note that the analysis provided by Reflect Audit is for informational purposes only and is not to be construed as financial advice or a solicitation to buy or sell assets. The contents of the audit reports and any associated materials are based on the information available at the time of the audit and do not represent a guarantee of the projects' future performance or success.

The findings within Reflect Audit reports reflect the security posture at the time of the audit according to the scope defined by the client. They should be utilized as one component of a comprehensive security strategy. Reflect Audit does not assume responsibility for any changes or updates to the project post-audit, nor does it hold liability for any actions taken based on the information provided.

Projects engaging with Reflect Audit are responsible for their own due diligence and for implementing any recommended security measures. The blockchain space is constantly evolving, and thus, the security landscape can shift rapidly. It is crucial for projects to maintain continuous vigilance and update their security protocols as needed.

By using Reflect Audit's services, clients acknowledge that smart contracts and blockchain technologies inherently carry risks, including but not limited to coding errors, vulnerabilities, and the potential for exploitation. While Reflect Audit strives to provide thorough and accurate assessments, we cannot unequivocally guarantee that audited contracts are free from all vulnerabilities.

Reflect Audit encourages projects to engage in ongoing security practices, such as maintaining a bug bounty program and conducting regular re-

audits, to strengthen and preserve the security integrity of their contracts over time.

Remember, investing in cryptocurrencies and blockchain projects comes with risks, and it is always recommended to consult with a financial advisor before making investment decisions. Reflect Audit's goal is to provide quality security assessments to enhance the overall security of the blockchain ecosystem.

About Reflect Audit

Reflect Audit brings smart solutions to the blockchain space. We're growing a system that's all about connection—tying together blockchain security, new token launches, NFT marketplaces, D apps and more. Our aim is to link up various services under one roof that's easy and quick for everyone to use.

Our team is worldwide and comes from all corners of the tech world. We're experts who care about making blockchain work better and safer for all. Reflect Audit offers detailed checks on smart contracts to catch any security issues before they become problems.

Want to know more? See what we do and reach out if you need a top-notch security check:
- Learn More: https://reflectaudit.com
- Book an Audit: https://t.me/MarkXODev

@reflectaudit

SECURING BLOCKCHAIN FUTURE WITH REFLECT AUDIT