DEFIMOON PROJECT
AUDIT AND
DEVELOPMENT

CONTACTS
HTTPS://DEFIMOON.ORG
AUDIT@DEFIMOON.ORG
TELEGRAM
TWITTER

REPORT SMART CONTRACT AUDIT

QUOTH.AI
TOKEN

FEBRUARY ,2022

AUDITOR
TEAM "DEFIMOON"

APPROVED BY
CYRILL MINYAEV, TEAM "DEFIMOON"

DEFIMOON

be secure

QUOTH.AI
FEBRUARY, 2022

DEFIMOON
be secure

REPORT
SMART CONTRACT AUDIT

AUDIT OF PROJECT

HTTPS://QUOTH.AI/

HTTPS://GITHUB.COM/QUOTH-AI/QUOTH-TOKEN/TREE/MAIN/CONTRACTS

Quoth Token

Disclaimer

This audit is not financial, investment, or any other kind of advice and could be used for informational purposes only. This report is not a substitute for doing your own research and due diligence should always be paid in full to any project. Defimoon is not responsible or liable for any loss, damage, or otherwise caused by reliance on this report for any purpose. Defimoon has based this audit report solely on the information provided by the audited party and on facts that existed before or during the audit being conducted. Defimoon is not responsible for any outcome, including changes done to the contract/contracts after the audit was published. This audit is fully objective and only discerns what the contract is saying without adding any opinion to it. Defimoon has no connection to the project other than the conduction of this audit and has no obligations other than to publish an objective report. Defimoon will always publish its findings regardless of the outcome of the findings. The audit only covers the subject areas detailed in this report and unless specifically stated, nothing else has been audited. Defimoon assumes that the provided information and materials were not altered, suppressed, or misleading. This report is published by Defimoon, and Defimoon has sole ownership of this report. Use of this report for any reason other than for informational purposes on the subjects reviewed in this report including the use of any part of this report is prohibited without the express written consent of Defimoon. In instances where an auditor or team member has a personal connection with the audited project, that auditor or team member will be excluded from viewing or impacting any internal communication regarding the specific audit.

Audit Information

Defimoon utilizes both manual and automated auditing approach to cover the most ground possible. We begin with generic static analysis automated tools to quickly assess the overall state of the contract. We then move to a comprehensive manual code analysis, which enables us to find security flaws that automated tools would miss. Finally, we conduct an extensive unit testing to make sure contract behaves as expected under stress conditions.

In our decision making process we rely on finding located via the manual code inspection and testing. If an automated tool raises a possible vulnerability, we always investigate it further manually to make a final verdict. All our tests are run in a special test environment which matches the "real world" situations and we utilize exact copies of the published or provided contracts.

While conducting the audit, the Defimoon security team uses best practices to ensure that the reviewed contracts are thoroughly examined against all angles of attack. This is done by evaluating the codebase and whether it gives rise to significant risks. During the audit, Defimoon assesses the risks and assigns a risk level to each section together with an explanatory comment.

Contents

Project information

Name	Quoth
Description	SEARCH AND AUTHENTICATE ANY NFT An all-chain NFT authentication oracle complete with AI and ML search, mint and bridge SDKs and APIs
Website	https://quoth.ai/
Twitter	https://twitter.com/Quoth_ai
Contact (Telegram)	@DeFiVlad
Token Name	Quoth Token
Token Short	$QUOTH
Total Supply	55,000,000
Token Decimals	18
Contract address	0xb025BC1675F6BE04eC528574712f268D99dB494d

Name	Quoth
Code Language	Solidity
Chain	Multichain

Audit overview

No security issues were found

The contract has no security issues. However some points should be taken into the consideration. In the current contract design, all tokens are minted to one address which holds all tokens. As Defimoon has not audited any other Quoth contract, we can’t confirm the IDO plans through contracts. Token transfer function is accessible to the contracts, inheriting from the reviewed Quoth contract. Even though it is not a security flaw and is not a vulnerability, keep in mind that the final verdict would depend on the use of this function by the inheriting contracts (which were not reviewed in the scope of this audit). Finally, the contract relies on the external Bot Protection contract with a closed codebase, which makes it impossible to review it. However, this protection contract was commissioned by the team and is used in other commercial projects, which increases the credibility. Also, the use of this contract could be disabled permanently if needed to. It is assumed that the Bot Protection contract is trustworthy, however, due to the impossibility to verify this assumption, a due diligence should be paid.

Those findings represent a "good to know while interacting with the project" information, but don’t directly damage the project in its current state, hence it’s up to the project team to resolve those issues if they feel the need to do so.

The UI/UX, logic, team, or tokenomics of the Quoth project were not reviewed by the Defimoon in the scope of this audition.

Please find the full report below for a complete understanding of the audit.

Application security checklist

Test	Result
Compiler errors	Passed
Possible delays in data delivery	Passed
Timestamp dependence	Passed
Integer Overflow and Underflow	Passed
Race Conditions and Reentrancy	Passed

Test	Result
DoS with Revert	Passed
DoS with block gas limit	Passed
Methods execution permissions	Passed
Economy model of the contract	Not Checked
Private user data leaks	Passed
Malicious Events Log	Passed
Scoping and Declarations	Passed
Uninitialized storage pointers	Passed
Arithmetic accuracy	Passed
Design Logic	Passed
Impact of the exchange rate	Not Checked
Oracle Calls	Not Checked
Cross-function race conditions	Passed
Safe OpenZeppelin contracts and implementation usage	Passed
Whitepaper-Website-Contract correlation	Not Checked
Front Running	Not Checked

Detailed audit information

Contract Programming

Test	Result
Solidity version not specified	Passed
Solidity version too old	Passed
Integer overflow/underflow	Passed
Function input parameters lack of check	Passed
Function input parameters check bypass	Passed
Function access control lacks management	Passed
Critical operation lacks event log	Passed
Human/contract checks bypass	Passed

Test	Result
Random number generation/use vulnerability	Passed
Fallback function misuse	Passed
Race condition	Passed
Logical vulnerability	Passed
Other programming issues	Passed

Code Specification

Test	Result
Visibility not explicitly declared	Passed
Variable storage location not explicitly declared	Passed
Use keywords/functions to be deprecated	Passed
Other code specification issues	Passed

Gas Optimization

Test	Result
Assert () misuse	Passed
High consumption ‘for/while’ loop	Passed
High consumption ‘storage’ storage	Passed
“Out of Gas” Attack	Passed

Business Risk

Test	Result
“Short Address” Attack	Passed
“Double Spend” Attack	Passed

Executive Summary

According to the standard audit assessment, client's solidity smart contract (Quoth project) is well-secured. The contract has successfully passed the audit.

Finding	ID	Severity	Status
Entire supply is minted to one address	#1	Low Risk	Open

Finding	ID	Severity	Status
External contract not available to review	#2	Low Risk	Acknowledged

Code Quality

Quoth project is well coded, the code is clear, concise and follows the best coding practices. However, it is always recommended to add comments to the code to make it easily understandable for a general user interested in the code.

Documentation

As mentioned above, it's recommended to write comments in the smart contract code, so that anyone can quickly understand the programming flow as well as code logic. The code was audited from the Quoth project GitHub repository, provided by the client and cross-checked with the deployed version of the contract.

Contract overview

Privileged executables

- `setBP`
- `toggleBP`
- `disableForeverBP`

Executables

- `_transfer`

Features

```
function setBP(address _bp) external onlyOwner {
        require(address(BP) == address(0), "QuothToken: BP already set");
        BP = BPContract(_bp);
}
```

Privileged
Allows to specify the address of the bot protection contract

```
function toggleBP() external onlyOwner {
        BPEnabled = !BPEnabled;
}
```

Privileged
Allows to turn bot protection on or off

```
function disableForeverBP() external onlyOwner {
        require(BPDisabledForever == false, "QuothToken BP already disabled forever");
        BPDisabledForever = true;
}
```

Privileged
Allows to disable the bot protection permanently. Could only be done once. It is impossible to turn it back on.

```
function _transfer(address sender, address recipient, uint256 amount) internal override(ERC20) {
        if (BPEnabled && !BPDisabledForever) BP.protect(sender, recipient, amount);
        super._transfer(sender, recipient, amount);
}
```

Allows to transfer assets from one address to another.

Findings

Static Analysis

Entire Supply Is Minted To One Address

Finding ID	**#1**
Severity	Low Risk
Status	Open
Location	QuothToken.sol > 14-22

```
constructor(
string memory name_,
string memory symbol_,
address recipient_,
uint256 amount_
) ERC20(name_, symbol_) {
        require(amount_ > 0, "QuothToken: Amount is zero");
        _mint(recipient_, amount_ * 10 ** decimals());
}
```

Description	**The token supply is minted to one address. The token website states that there will be an IDO with vested token holders [1]. As no other contract has been provided Defimoon assumes that an Externally Owned Address will handle the token supply and distribution.**
Recomendation	It is suggested to deploy a specialized smart contract to handle the vested token distribution
Resolution	N/A

[1] https://quoth.ai/0x/Links/QuothVestingExpanded.pdf

External contract not available to review

Finding ID	**#2**
Severity	Low Risk
Status	Acknowledged
Location	QuothToken.sol > 39

```
if (BPEnabled && !BPDisabledForever) BP.protect(sender, recipient, amount);
```

Description	**QuothToken contract relies on an external contract with a closed codebase which makes it impossible to review and audit.**
Recomendation	Some companies are unwilling to disclose their code and make it open-source to protect their commercial rights. While this is an absolutely normal practice, situations like this should be handled with care.

Description	QuothToken contract relies on an external contract with a closed codebase which makes it impossible to review and audit.
Resolution	QuothToken developers implemented the ability to disable the external contract permanently, thus adding a layer of protection in case an issue might emerge

On-chain Analysis

Contract Address	0xb025BC1675F6BE04eC528574712f268D99dB494d
Creator Address	0×56B5bB4a1B7B78358193C00dD4Bc7222A300127e
Creating TxT hash	0×87ac6a61f52e7d9fc0ed528a4d769a58e755a251e1cc21b144a665969c362

Contract code is verified on chain and is the exact match with the provided codebase.

Imported contracts and dependencies

OpenZeppelin ERC20	Industry standard implementation of the ERC20 specification
OpenZeppelin Ownable	Industry standard implementation of the owner privileged features
Bot Protection	Bot protection mechanism

Conclusion

The codebase of the Quoth Project has passed the audit successfully and can be considered a "Well-Secured" application. The code is well-written, clear and follows the best security practices. On-chain information matches the provided information.

However, due to the nature of the application and given the risks connected with the decentralized finance, we can provide no guarantees on the future outcomes and project operation. We have used all the latest static tools and manual analysis to cover the greatest possible amount of test cases. Smart contracts within the scope were manually reviewed and analyzed with static analysis tools. Smart contracts high-level description of functionality and security was presented in the Application security checklist section of the report.

Audit report contains all found security vulnerabilities and other issues found in the reviewed code.

Security status of the reviewed codebase is "Well-Secured".

Methodology

We like to work with a transparent process and make our reviews a collaborative effort. The goal of our security audits is to improve the quality of systems we review and aim for sufficient remediation to help protect users. The following is the methodology we use in our security audit process.

Manual Code Review

In manually reviewing the code, we look for any potential issues with code logic, error handling, protocol and header parsing, cryptographic errors, and random number generators. We also watch for areas where more defensive programming could reduce the risk of future mistakes and speed up future audits. Although our primary focus is on the in-scope code, we examine dependency code and behavior when it is relevant to a particular line of investigation.

Vulnerability Analysis

Our audit techniques include manual code analysis, user interface interaction, and whitebox penetration testing. We look at the project's web site to get a high-level understanding of what functionality the software under review provides. We then meet with the developers to gain an appreciation of their vision of the software. We install and use the relevant software, exploring the user interactions and roles. While we do this, we brainstorm threat models and attack surfaces. We read design documentation, review other audit results, search for similar projects, examine source code dependencies, review open issue tickets, and investigate details other than the implementation.

Documenting Results

We follow a conservative, transparent process for analyzing potential security vulnerabilities and seeing them through successful remediation. Whenever a potential issue is discovered, we immediately create an Issue entry for it in this document, even though we have not yet verified the feasibility and impact of the issue. This process is conservative because we document our suspicions early even if they are later shown to not represent exploitable vulnerabilities. We follow a process of first documenting the suspicion with unresolved questions, then confirming the issue through code analysis, live experimentation, or automated tests. Code analysis is the most tentative, and we strive to provide test code, log captures, or screenshots demonstrating our confirmation. After this we analyze the feasibility of an attack in a live system to make a final decision.

Suggested Solutions

We search for immediate mitigations that live deployments can take, and finally we suggest the requirements for remediation engineering for future releases. The mitigation and remediation recommendations should be scrutinized by the developers and deployment engineers, and successful mitigation and remediation is an ongoing collaborative process after we deliver our report, and before the details are made public.

Appendix A — Risk Ratings

Risk Rating	Description
High Risk	A fatal vulnerability that can cause the loss of all Tokens / Funds.
Medium Risk	A vulnerability that can cause the loss of some Tokens / Funds.
Low Risk	A vulnerability which can cause the loss of protocol functionality.
Informational	Non-security issues such as functionality, style, and convention.

Appendix B — Finding Statuses

Status	Description
Closed	Contracts were modified to permanently resolve the finding.
Mitigated	The finding was resolved by other methods such as revoking contract ownership. The issue may require monitoring, for example in the case of a time lock.
Partially Closed	Contracts were updated to fix the issue in some parts of the code.
Partially Mitigated	Fixed by project specific methods which cannot be verified on chain. Examples include compounding at a given frequency.
Acknowledged	Project team is made aware of the finding.
Open	The finding was not addressed.