

HEX NINJA AUDITS



Security Assessment

Lunar Token

October 4, 2022

Table of Contents

1 Assessment Summary

2 Technical Findings Summary

3 Project Overview

3.1 Token Summary

3.2 Risk Analysis Summary

3.3 Main Contract Assessed

4 Smart Contract Risk Checks

4.1 Mint Check

4.2 Fees Check

4.3 Blacklist Check

4.4 MaxTx Check

4.5 Pause Trade Check

5 Contract Ownership

6 Liquidity Ownership

7 KYC Check

8 Smart Contract Vulnerability Checks

8.1 Smart Contract Vulnerability Details

8.2 Smart Contract Inheritance Details

8.3 Smart Contract Privileged Functions

9 Assessment Results and Notes(Important)

10 Social Media Check(Informational)

11 Technical Findings Details

12 Disclaimer



Assessment Summary

This report has been prepared for Lunar Token on the Binance Smart Chain network. CFGNINJA provides both client-centered and user-centered examination of the smart contracts and their current status when applicable. This report represents the security assessment made to find issues and vulnerabilities on the source code along with the current liquidity and token holder statistics of the protocol.

A comprehensive examination has been performed, utilizing Cross Referencing, Static Analysis, In-House Security Tools, and line-by-line Manual Review.






The auditing process pays special attention to the following considerations:

- Testing the smart contracts against both common and uncommon attack vectors.
- Inspecting liquidity and holders statistics to inform the current status to both users and client when applicable.
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Verifying contract functions that allow trusted and/or untrusted actors to mint, lock, pause, and transfer assets.
- Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders
- Thorough line-by-line manual review of the entire codebase by industry experts.








Technical Findings Summary

Classification of Risk

Severity	Description
 Critical	Risks are those that impact the safe functioning of a platform and must be addressed before launch. Users should not invest in any project with outstanding critical risks.
 Major	Risks can include centralization issues and logical errors. Under specific circumstances, these major risks can lead to loss of funds and/or control of the project.
 Medium	Risks may not pose a direct risk to users' funds, but they can affect the overall functioning of a platform
 Minor	Risks can be any of the above but on a smaller scale. They generally do not compromise the overall integrity of the Project, but they may be less efficient than other solutions.
 Informational	Errors are often recommended to improve the code's style or certain operations to fall within industry best practices. They usually do not affect the overall functioning of the code.

Findings

Severity	Found	Pending	Resolved
 Critical	0	0	0
 Major	0	0	0
 Medium	0	0	0
 Minor	0	0	0
 Informational	1	0	1
Total	1	0	1



Project Overview

Token Summary

Parameter	Result
Address	0xc1A59a17F87ba6651Eb8E8F707db7672647c45bD
Name	Lunar
Token Tracker	Lunar (LNR)
Decimals	18
Supply	1000000000
Platform	Binance Smart Chain
compiler	v0.8.17+commit.8df45f5f
Contract Name	Lunar
Optimization	Yes with 200 runs
LicenseType	MIT
Language	Solidity
Codebase	https://bscscan.com/address/0xc1A59a17F87ba6651Eb8E8F707db7672647c45bD#code
Payment Tx	0x81f3fcd1e9f0c6aee2d9a516aa8ebaa9a1d0e0b6fb4765d28060b6b638dd577



Project Overview

Risk Analysis Summary

Parameter	Result
Buy Tax	6%
Sale Tax	6%
Is honeypot?	Clean
Can edit tax?	Yes
Is anti whale?	Yes
Is blacklisted?	Yes
Is whitelisted?	Yes
Holders	Clean
Security Score	95/100
Auditor Score	99/100
Confidence Level	High

The following quick summary it's added to the project overview; however, there are more details about the audit and its results. Please read every detail.



Main Contract Assessed Contract Name

Name	Contract	Live
Lunar	0xc1A59a17F87ba6651Eb8E8F707db7672647c45bD	Yes

TestNet Contract Assessed Contract Name

Name	Contract	Live
Lunar	0xA8CA720d120870f9cb6F99C2d73707849DDD1f83	Yes

Solidity Code Provided

SolID	File Sha-1	FileName
ERC20	4d4cadbb6d32f5ce4903ddeac89347d718f9172a	ERC20.sol



Mint Check

The project owners of Lunar do not have a mint function in the contract, owner cannot mint tokens after initial deploy.

The Project has a Total Supply of 1000000000 and cannot mint any more than the Max Supply.

Mint Notes:

Auditor Notes: Customer has a mint compliance and cannot mint more than the total supply.

Project Owner Notes:



Fees Check

The project owners of Lunar do not have the ability to set fees higher than 25% .

The team May have fees defined; however, they can't set those fees higher than 25% or may not be able to configure the same.

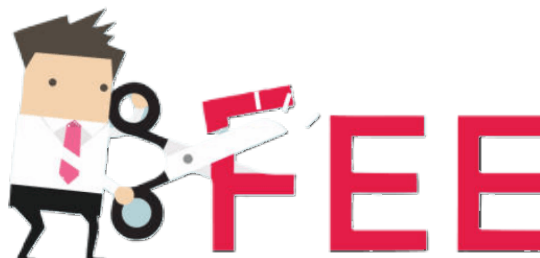
Tax Fee Notes:

Auditor Notes: The contract currently has 6% buy and 6% sale taxes.

Project Owner Notes: .



Fees can be changed up to a maximum of 25%



Blacklist Check

The project owners of Lunar have the ability to Blacklist holders from transferring their tokens.

We recommend the Team be careful with a blacklist function as this can prevent a holder from buying/selling/transferring their assets. Malicious or compromised owners can trap contracts relying on tokens with a blacklist

Blacklist Notes:

Auditor Notes: The contract has a blacklist function; however, this is done per wallet basis.

Project Owner Notes: Project Owner states, ' We need the ability to comply with potential international sanctions on certain wallet addresses, and have the ability to quarantine malicious hackers that could harm the integrity of the protocol.'



MaxTx Check

The Project Owners of Lunar cannot set max tx amount

The Team allows any investors to swap, transfer or sell their total amount if needed.

MaxTX Notes:

Auditor Notes: There is a Max Wallet function; however, it cannot go below the desired threshold for safety reasons.

Project Owner Notes: The project Owner states: 'Our measures against any one wallet controlling the system means that wallets are limited to 1% of the max supply.'

Project Has No MaxTX



Pause Trade Check

The Project Owners of Lunar can stop or pause trading

We recommend the Team only allow Open Trade and never use Stop Trade, as this will be catastrophic for the Project and Investors.

We recommend the Team create a new contract without the stop trade function.

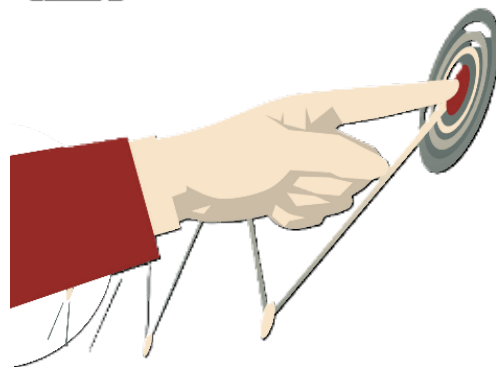
Pause Trade Notes:

Auditor Notes: The project Owner can start and stop trade at any time, and we have recommended the Team review this.

Project Owner Notes: Project Owner States: 'Because the contract is upgradeable, if we make a mistake during a future upgrade that starts propagating incorrect results for transactions, we need to potentially be able to stop the problem from cascading while we fix the problem and deploy an upgraded contract.'



Owner can pause trading



Contract Ownership

The contract ownership of Lunar is not currently renounced. The ownership of the contract grants special powers to the protocol creators, making them the sole addresses that can call sensible ownable functions that may alter the state of the protocol.

The current owner is the address
0x8C1DF8d7BcBE1395Ef66508F76a8732EaB65FBeE
which can be viewed:
[HERE](#)

The owner wallet has the power to call the functions displayed on the privileged functions chart below, if the owner's wallet is compromised, they could exploit these privileges.

We recommend the team renounce ownership at the right time, if possible, or gradually migrate to a timelock with governing functionalities regarding transparency and safety considerations.

We recommend the team use a Multisignature Wallet if the contract is not going to be renounced; this will give the team more control over the contract.



Liquidity Ownership

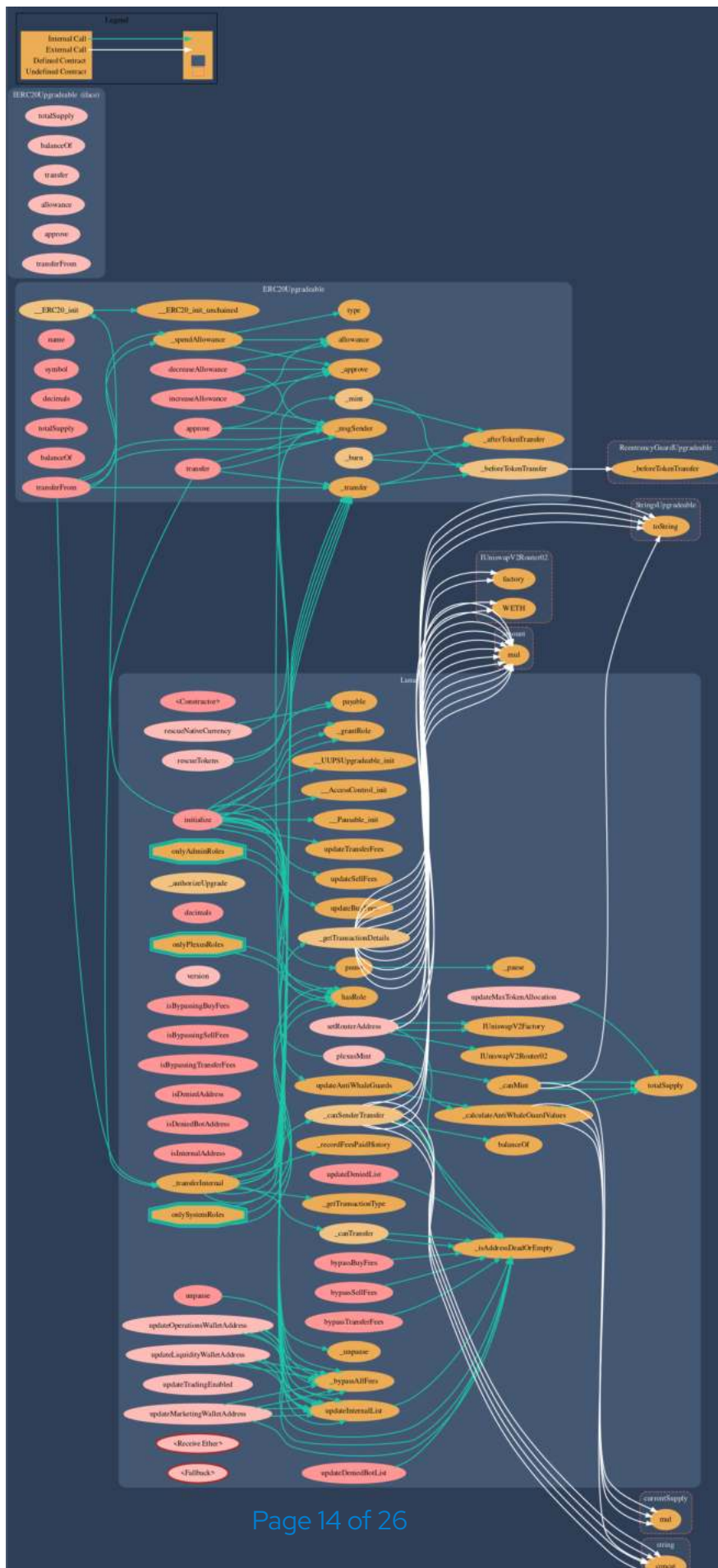
Most of the liquidity is currently locked; the lock can be seen here:

Liquidity Locker Link can be viewed from:
[HERE](#)



Call Graph

The contract for Lunar has the following call graph structure.



KYC Information

The Project Owners of Lunar have provided KYC Documentation.

KYC Certificated can be found on the Following:
KYC Data

KYC Information Notes:

Auditor Notes: Customer is KYC by Dessert Finance

Project Owner Notes:



Smart Contract Vulnerability Checks

ID	Severity	Name	File	location
SWC-100	Pass	Function Default Visibility	ERC20.sol	L: 0 C: 0
SWC-101	Pass	Integer Overflow and Underflow.	ERC20.sol	L: 0 C: 0
SWC-102	Pass	Outdated Compiler Version file.	ERC20.sol	L: 0 C: 0
SWC-103	Low	A floating pragma is set.	ERC20.sol	L: 2 C: 1
SWC-104	Pass	Unchecked Call Return Value.	ERC20.sol	L: 0 C: 0
SWC-105	Pass	Unprotected Ether Withdrawal.	ERC20.sol	L: 0 C: 0
SWC-106	Pass	Unprotected SELFDESTRUCT Instruction	ERC20.sol	L: 0 C: 0
SWC-107	Pass	Read of persistent state following external call.	ERC20.sol	L: 0 C: 0
SWC-108	Pass	State variable visibility is not set..	ERC20.sol	L: 428 C: 29
SWC-109	Pass	Uninitialized Storage Pointer.	ERC20.sol	L: 0 C: 0
SWC-110	Pass	Assert Violation.	ERC20.sol	L: 0 C: 0
SWC-111	Pass	Use of Deprecated Solidity Functions.	ERC20.sol	L: 0 C: 0
SWC-112	Pass	Delegate Call to Untrusted Callee.	ERC20.sol	L: 0 C: 0



ID	Severity	Name	File	location
SWC-113	Pass	Multiple calls are executed in the same transaction.	ERC20.sol	L: 0 C: 0
SWC-114	Pass	Transaction Order Dependence.	ERC20.sol	L: 0 C: 0
SWC-115	Pass	Authorization through tx.origin.	ERC20.sol	L: 0 C: 0
SWC-116	Pass	A control flow decision is made based on The block.timestamp environment variable.	ERC20.sol	L: 0 C: 0
SWC-117	Pass	Signature Malleability.	ERC20.sol	L: 0 C: 0
SWC-118	Pass	Incorrect Constructor Name.	ERC20.sol	L: 0 C: 0
SWC-119	Pass	Shadowing State Variables.	ERC20.sol	L: 0 C: 0
SWC-120	Pass	Potential use of block.number as source of randomness.	ERC20.sol	L: 421 C: 12, L: 496 C: 22, L: 497 C: 28
SWC-121	Pass	Missing Protection against Signature Replay Attacks.	ERC20.sol	L: 0 C: 0
SWC-122	Pass	Lack of Proper Signature Verification.	ERC20.sol	L: 0 C: 0
SWC-123	Pass	Requirement Violation.	ERC20.sol	L: 0 C: 0
SWC-124	Pass	Write to Arbitrary Storage Location.	ERC20.sol	L: 0 C: 0
SWC-125	Pass	Incorrect Inheritance Order.	ERC20.sol	L: 0 C: 0
SWC-126	Pass	Insufficient Gas Griefing.	ERC20.sol	L: 0 C: 0



ID	Severity	Name	File	location
SWC-127	Pass	Arbitrary Jump with Function Type Variable.	ERC20.sol	L: 0 C: 0
SWC-128	Pass	DoS With Block Gas Limit.	ERC20.sol	L: 0 C: 0
SWC-129	Pass	Typographical Error.	ERC20.sol	L: 0 C: 0
SWC-130	Pass	Right-To-Left-Override control character (U+202E).	ERC20.sol	L: 0 C: 0
SWC-131	Pass	Presence of unused variables.	ERC20.sol	L: 0 C: 0
SWC-132	Pass	Unexpected Ether balance.	ERC20.sol	L: 0 C: 0
SWC-133	Pass	Hash Collisions with Multiple Variable Length Arguments.	ERC20.sol	L: 0 C: 0
SWC-134	Pass	Message call with hardcoded gas amount.	ERC20.sol	L: 0 C: 0
SWC-135	Pass	Code With No Effects (Irrelevant/Dead Code).	ERC20.sol	L: 0 C: 0
SWC-136	Pass	Unencrypted Private Data On-Chain.	ERC20.sol	L: 0 C: 0

We scan the contract for additional security issues using MYTHX and industry-standard security scanning tools.



Smart Contract Vulnerability Details

SWC-103 - Floating Pragma.

CWE-664: Improper Control of a Resource Through its Lifetime.

References:

Description:

Contracts should be deployed with the same compiler version and flags that they have been tested with thoroughly. Locking the pragma helps to ensure that contracts do not accidentally get deployed using, for example, an outdated compiler version that might introduce bugs that affect the contract system negatively.

Remediation:

Lock the pragma version and also consider known bugs (<https://github.com/ethereum/solidity/releases>) for the compiler version that is chosen.

Pragma statements can be allowed to float when a contract is intended for consumption by other developers, as in the case with contracts in a library or EthPM package. Otherwise, the developer would need to manually update the pragma in order to compile locally.

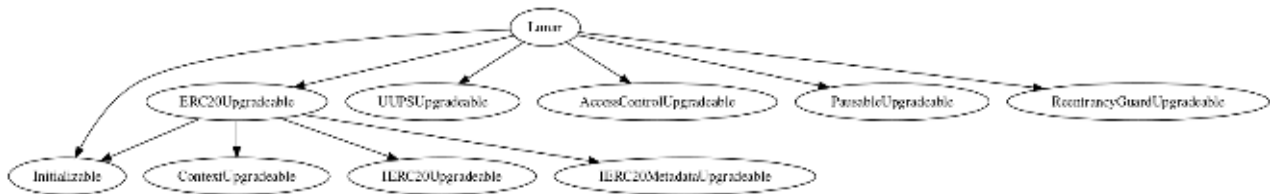
References:

Ethereum Smart Contract Best Practices - Lock pragmas to specific compiler version.



Inheritance

The contract for Lunar has the following inheritance structure.



Privileged Functions (onlyOwner)

Function Name	Parameters	Visibility
updateTransferFees	none	public
updateTradingEnabled	none	external
updateSellFees	none	external
updateOperationsWalletAddress	none	external
updateMaxTokenAllocation	none	external
pause	none	public
plexusMint	none	public
rescueNativeCurrency	none	public
rescueTokens	none	public





Assessment Results

- Contract is a Proxy Contract and it's implementation is under
0xb8448630a74ad7E871265ae661b8c3e470F7b5a4
- Lunar has been a very successful project; while this contract is a proxy contract and can be upgraded, the project team has ensured its safety.
- No high-risk Exploits/Vulnerabilities Were Found in the Source Code.

Audit Passed



LNR-02 | Function Visibility Optimization.

Category	Severity	Location	Status
Gas Optimization	 Informational	ERC20.sol: 424,13	 Acknowledged

Description

The following functions are declared as public and are not invoked in any of the contracts contained within the projects scope:

Function Name	Parameters	Visibility
name		public
symbol		public
decimals		public
pause		public
unpause		public
updateBuyFees		public
updateSaleFees		public

The functions that are never called internally within the contract should have external visibility

Remediation

We advise that the function's visibility specifiers are set to external, and the array-based arguments change their data location from memory to calldata, optimizing the gas cost of the function.

References:

external vs public best practices.



Social Media Checks

Social Media	URL	Result
Twitter	https://twitter.com/lnr	Pass
Instagram	https://www.instagram.com/lnrdefi/	Pass
Website	https://lunar.io/	Pass
Telegram	https://t.me/lnrDefi	Pass

We recommend to have 3 or more social media sources including a completed working websites.

Social Media Information Notes:

Auditor Notes: undefined

Project Owner Notes: Project also have discord under <https://discord.gg/lnr>



Appendix

Finding Categories

Centralization / Privilege

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

Gas Optimization

Gas Optimization findings do not affect the functionality of the code but generate different, more optimal EVM 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 how block.timestamp works.

Control Flow

Control Flow findings concern the access control imposed on functions, such as owner-only functions being 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 may result in a vulnerability.

Coding Style

Coding Style findings usually do not affect the generated byte-code but rather comment on how to make the 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 requirements on the input variables than a setter function.

Coding Best Practices

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



Disclaimer

CFGNINJA has conducted an independent security assessment to verify the integrity of and highlight any vulnerabilities or errors, intentional or unintentional, that may be present in the reviewed code for the scope of this assessment. This report does not constitute agreement, acceptance, or advocacy for the Project, and users relying on this report should not consider this as having any merit for financial advice in any shape, form, or nature. The contracts audited do not account for any economic developments that the Project in question may pursue, and the veracity of the findings thus presented in this report relate solely to the proficiency, competence, aptitude, and discretion of our independent auditors, who make no guarantees nor assurance that the contracts are entirely free of exploits, bugs, vulnerabilities or deprecation of technologies.

All information provided in this report does not constitute financial or investment advice, nor should it be used to signal that any persons reading this report should invest their funds without sufficient individual due diligence, regardless of the findings presented. Information is provided 'as is, and CFGNINJA is under no covenant to audited completeness, accuracy, or solidity of the contracts. In no event will CFGNINJA or its partners, employees, agents, or parties related to the provision of this audit report be liable to any parties for, or lack thereof, decisions or actions with regards to the information provided in this audit report.

The assessment services provided by CFGNINJA are subject to dependencies and are under continuing development. You agree that your access or use, including but not limited to any services, reports, and materials, will be at your sole risk on an as-is, where-is, and as-available basis. Cryptographic tokens are emergent technologies with high levels of technical risk and uncertainty. The assessment reports could include false positives, negatives, and unpredictable results. The services may access, and depend upon, multiple layers of third parties.

