

Assure DeFi™

The Verification **Gold Standard**™



Security Assessment **Lynx Tech Staking**

December 14, 2023

Audit Status: Pass

Audit Edition: Advance



Project Overview

Token Summary

Parameter	Result
Address	0x
Name	Lynx Tech
Token Tracker	Lynx Tech (\$Lynx)
Decimals	0
Supply	0
Platform	Ethereum
compiler	v0.8.21+commit.d9974bed
Contract Name	LynxStaking
Optimization	Yes with 200 runs
LicenseType	MIT
Language	Solidity
Codebase	https://etherscan.io/address/0x#code
Payment Tx	Corporate

Main Contract Assessed

Contract Name

Name	Contract	Live
Lynx Tech	0x	No

TestNet Contract Assessed

Contract Name

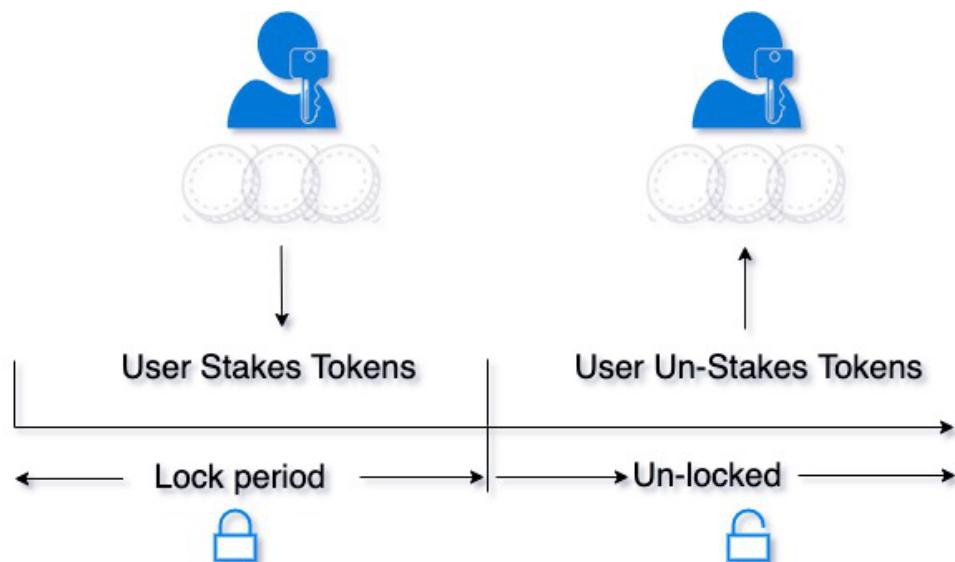
Name	Contract	Live
Lynx Tech	0xa16Cd338D152DF0373c5FA4212faD4069e59D8a3	No

Solidity Code Provided

Solid ID	File Sha-1	FileName
Staking	ad6b3b27d6f7ef70fefcccd90c664d599c926a13	LynxFlatStaking.sol
Staking	ac4116c40704bb9f4236f78e306b4608058d61a1	VaultFlat.sol
Staking		
Staking		

What is a Staking Contract

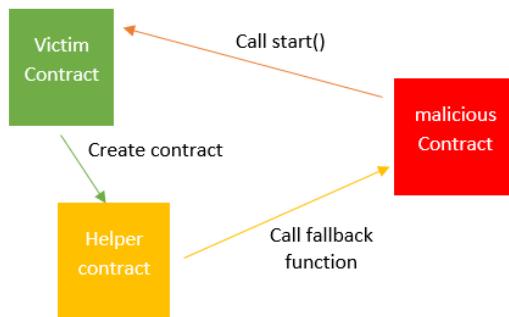
A smart contract which allows users to stake and un-stake a specified ERC20 token. Staked tokens are locked for a specific length of time (set by the contract owner at the outset). Once the time period has elapsed, the user can remove their tokens again.



The Project Owners of Lynx Tech 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**



Smart Contract Vulnerability Checks

The Smart Contract Weakness Classification Registry (SWC Registry) is an implementation of the weakness classification scheme proposed in EIP-1470. It is loosely aligned to the terminologies and structure used in the Common Weakness Enumeration (CWE) while overlaying a wide range of weakness variants that are specific to smart contracts.

ID	Severity	Name	File	location
SWC-100	Pass	Function Default Visibility	LynxFlatStaking.sol	L: 0 C: 0
SWC-101	Pass	Integer Overflow and Underflow.	LynxFlatStaking.sol	L: 0 C: 0
SWC-102	Pass	Outdated Compiler Version file.	LynxFlatStaking.sol	L: 0 C: 0
SWC-103	Pass	A floating pragma is set.	LynxFlatStaking.sol	L: 0 C: 0
SWC-104	Pass	Unchecked Call Return Value.	LynxFlatStaking.sol	L: 0 C: 0
SWC-105	Pass	Unprotected Ether Withdrawal.	LynxFlatStaking.sol	L: 0 C: 0
SWC-106	Pass	Unprotected SELFDESTRUCT Instruction	LynxFlatStaking.sol	L: 0 C: 0
SWC-107	Pass	Read of persistent state following external call.	LynxFlatStaking.sol	L: 0 C: 0
SWC-108	Pass	State variable visibility is not set..	LynxFlatStaking.sol	L: 0 C: 0
SWC-109	Pass	Uninitialized Storage Pointer.	LynxFlatStaking.sol	L: 0 C: 0
SWC-110	Pass	Assert Violation.	LynxFlatStaking.sol	L: 0 C: 0
SWC-111	Pass	Use of Deprecated Solidity Functions.	LynxFlatStaking.sol	L: 0 C: 0
SWC-112	Pass	Delegate Call to Untrusted Callee.	LynxFlatStaking.sol	L: 0 C: 0
SWC-113	Pass	Multiple calls are executed in the same transaction.	LynxFlatStaking.sol	L: 0 C: 0
SWC-114	Pass	Transaction Order Dependence.	LynxFlatStaking.sol	L: 0 C: 0

ID	Severity	Name	File	location
SWC-115	Pass	Authorization through tx.origin.	LynxFlatStaking.sol	L: 0 C: 0
SWC-116	Pass	A control flow decision is made based on The block.timestamp environment variable.	LynxFlatStaking.sol	L: 0 C: 0
SWC-117	Pass	Signature Malleability.	LynxFlatStaking.sol	L: 0 C: 0
SWC-118	Pass	Incorrect Constructor Name.	LynxFlatStaking.sol	L: 0 C: 0
SWC-119	Pass	Shadowing State Variables.	LynxFlatStaking.sol	L: 0 C: 0
SWC-120	Pass	Potential use of block.number as source of randomness.	LynxFlatStaking.sol	L: 0 C: 0
SWC-121	Pass	Missing Protection against Signature Replay Attacks.	LynxFlatStaking.sol	L: 0 C: 0
SWC-122	Pass	Lack of Proper Signature Verification.	LynxFlatStaking.sol	L: 0 C: 0
SWC-123	Pass	Requirement Violation.	LynxFlatStaking.sol	L: 0 C: 0
SWC-124	Pass	Write to Arbitrary Storage Location.	LynxFlatStaking.sol	L: 0 C: 0
SWC-125	Pass	Incorrect Inheritance Order.	LynxFlatStaking.sol	L: 0 C: 0
SWC-126	Pass	Insufficient Gas Griefing.	LynxFlatStaking.sol	L: 0 C: 0
SWC-127	Pass	Arbitrary Jump with Function Type Variable.	LynxFlatStaking.sol	L: 0 C: 0
SWC-128	Pass	DoS With Block Gas Limit.	LynxFlatStaking.sol	L: 0 C: 0
SWC-129	Pass	Typographical Error.	LynxFlatStaking.sol	L: 0 C: 0
SWC-130	Pass	Right-To-Left-Override control character (U+202E).	LynxFlatStaking.sol	L: 0 C: 0
SWC-131	Pass	Presence of unused variables.	LynxFlatStaking.sol	L: 0 C: 0
SWC-132	Pass	Unexpected Ether balance.	LynxFlatStaking.sol	L: 0 C: 0

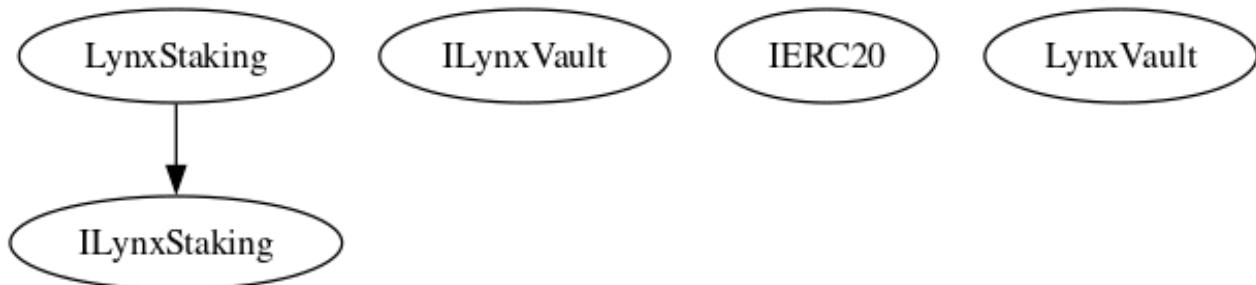
ID	Severity	Name	File	location
SWC-133	Pass	Hash Collisions with Multiple Variable Length Arguments.	LynxFlatStaking.sol	L: 0 C: 0
SWC-134	Pass	Message call with hardcoded gas amount.	LynxFlatStaking.sol	L: 0 C: 0
SWC-135	Pass	Code With No Effects (Irrelevant/Dead Code).	LynxFlatStaking.sol	L: 0 C: 0
SWC-136	Pass	Unencrypted Private Data On-Chain.	LynxFlatStaking.sol	L: 0 C: 0

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

Inheritance

The contract for Lynx Tech has the following inheritance structure.

The Project has a Total Supply of 0



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.
🟠 High	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
⚡ Low	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.
ℹ️ 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
🟠 High	0	0	0
🟡 Medium	0	0	0
⚡ Low	0	0	0
ℹ️ Informational	0	0	0
Total	0	0	0

Social Media Checks

Social Media	URL	Result
Twitter	@LynxTech_io	Pass
Other	https://linkedin.com/company/lynx-tech-inc/	Pass
Website	https://lynxtech.io/	Pass
Telegram	https://t.me/LYNX_Tech_Official	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:



Audit Result

Final Audit Score

Review	Score
Security Score	90
Auditor Score	90

The Following Score System Has been Added to this page to help understand the value of the audit, the maximum score is 100, however to attain that value the project must pass and provide all the data needed for the assessment. Our Passing Score has been changed to 85 Points for a higher standard, if a project does not attain 85% is an automatic failure. Read our notes and final assessment below.

Audit Passed



Assessment Results

Important Notes:

- No vulnerabilities or exploits were detected at the time of the audit.¹
- The following code is a staking contract.¹
- We have also audited the vault contract as part of this audit.

**Auditor Score =90
Audit Passed**



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 require statements 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

Assure Defi 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 depreciation 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 Assure Defi is under no covenant to audited completeness, accuracy, or solidity of the contracts. In no event will Assure Defi 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 Assure Defi 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.

