CFG NINJA AUDITS

Security Assessment

Pinksale Subscription

August 3, 2022

Table of Contents

Audit Summary

This report has been prepared for Pinksale Subscription 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.

Project Overview

Token Summary

Parameter	Result
Address	0x5e45ee804431343c559fc7382ae7f6c1d6873ecc
Name	Pinksale Subscription
Token Tracker	Pinksale Subscription ()
Decimals	
Supply	
Platform	Binance Smart Chain
compiler	v0.8.15+commit
Contract Name	Subscription
Optimization	Yes with 200 runs
LicenseType	MIT
Language	Solidity
Codebase	https://bscscan.com/ address/0x5e45ee804431343c559fc7382ae7f6c1d6873ecc
Payment Tx	

Main Contract Assessed
Contract Name

Name	Contract	Live
Pinksale Subscription	0x5e45ee804431343c559fc7382ae7f6c1d6873ecc	Yes

TestNet Contract Assessed
Contract Name

Name	Contract	Live
Pinksale Subscription	0xdfaae46ee412395db23e844b21f7c8a1f55b7012	Yes

Solidity Code Provided

SolID	File Sha-1	FileName
Subscription	6999446890e28a89241f741c7af896433a3ff87c	Subscription.sol
Subscription		
Subscription		
Subscription	undefined	
Subscription	undefined	

Smart Contract Vulnerability Checks

Vulnerability	Automatic Scan	Manual Scan	Result
Unencrypted Private Data On-Chain	Complete	Complete	Low / No Risk
Code With No Effects	Complete	Complete	Low / No Risk
Message call with hardcoded gas amount	Complete	Complete	Low / No Risk
Hash Collisions With Multiple Variable Length Arguments	Complete	Complete	Low / No Risk
Unexpected Ether balance	Complete	Complete	Low / No Risk
Presence of unused variables	Complete	Complete	Low / No Risk
Right-To-Left-Override control character (U+202E)	Complete	Complete	Low / No Risk
Typographical Error	Complete	Complete	Low / No Risk
DoS With Block Gas Limit	Complete	Complete	Low / No Risk
Arbitrary Jump with Function Type Variable	Complete	Complete	Low / No Risk
Insufficient Gas Griefing	Complete	Complete	Low / No Risk
Incorrect Inheritance Order	Complete	Complete	Low / No Risk
Write to Arbitrary Storage Location	Complete	Complete	Low / No Risk
Requirement Violation	Complete	Complete	Low / No Risk
Missing Protection against Signature Replay Attacks	Complete	Complete	Low / No Risk

Contract Ownership

The contract ownership of Pinksale Subscription 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 which can be viewed from:
HERE

The owner wallet has the power to call the functions displayed on the priviliged functions chart below, if the owner wallet is compromised this privileges could be exploited.

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

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

KYC Information

The Project Onwers of Pinksale Subscription has provided KYC Documentation.

KYC Certificated can be found on the Following:
KYC Data

KYC Information Notes:

Auditor Notes:

Project Owner Notes: Customer is KYC

Mythx Security Summary Checks

ID	Severity	Name	File	location
SWC-100	Pass	Function Default Visibility	Subscription.sol	L: 0 C: 0
SWC-101	Pass	Integer Overflow and Underflow.	Subscription.sol	L: 0 C: 0
SWC-102	Pass	Outdated Compiler Version file.	Subscription.sol	L: 0 C: 0
SWC-103	Low	A floating pragma is set.	Subscription.sol	L: 5 C: 0
SWC-104	Pass	Unchecked Call Return Value.	Subscription.sol	L: 0 C: 0
SWC-105	Pass	Unprotected Ether Withdrawal.	Subscription.sol	L: 0 C: 0
SWC-106	Pass	Unprotected SELFDESTRUCT Instruction	Subscription.sol	L: 0 C: 0
SWC-107	Low	Read of persistent state following external call.	Subscription.sol	L: 0 C: 0
SWC-108	Pass	State variable visibility is not set..	Subscription.sol	L: 0 C: 0
SWC-109	Pass	Uninitialized Storage Pointer.	Subscription.sol	L: 0 C: 0
SWC-110	Pass	Assert Violation.	Subscription.sol	L: 0 C: 0
SWC-111	Pass	Use of Deprecated Solidity Functions.	Subscription.sol	L: 0 C: 0
SWC-112	Pass	Delegate Call to Untrusted Callee.	Subscription.sol	L: 0 C: 0
SWC-113	Low	Multiple calls are executed in the same transaction.	Subscription.sol	L: 0 C: 0

ID	Severity	Name	File	location
SWC-114	Pass	Transaction Order Dependence.	Subscription.sol	L: 0 C: 0
SWC-115	Pass	Authorization through tx.origin.	Subscription.sol	L: 474 C: 15
SWC-116	Low	A control flow decision is made based on The block.timestamp environment variable.	Subscription.sol	L: 0 C: 0
SWC-117	Pass	Signature Malleability.	Subscription.sol	L: 0 C: 0
SWC-118	Pass	Incorrect Constructor Name.	Subscription.sol	L: 0 C: 0
SWC-119	Pass	Shadowing State Variables.	Subscription.sol	L: 0 C: 0
SWC-120	Pass	Potential use of block.number as source of randonmness.	Subscription.sol	L: 0 C: 0
SWC-121	Pass	Missing Protection against Signature Replay Attacks.	Subscription.sol	L: 0 C: 0
SWC-122	Pass	Lack of Proper Signature Verification.	Subscription.sol	L: 0 C: 0
SWC-123	Pass	Requirement Violation.	Subscription.sol	L: 0 C: 0
SWC-124	Pass	Write to Arbitrary Storage Location.	Subscription.sol	L: 0 C: 0
SWC-125	Pass	Incorrect Inheritance Order.	Subscription.sol	L: 0 C: 0
SWC-126	Pass	Insufficient Gas Griefing.	Subscription.sol	L: 0 C: 0
SWC-127	Pass	Arbitrary Jump with Function Type Variable.	Subscription.sol	L: 0 C: 0
SWC-128	Pass	DoS With Block Gas Limit.	Subscription.sol	L: 0 C: 0

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

We scan the contract for additional security issues using MYTHX and industry standard security scanning tool

Security Check Details Page

SWC-103 - Floating Pragma.

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

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.

SWC-107 - Reentrancy.

CWE-841: Improper Enforcement of Behavioral Workflow.

Description:

One of the major dangers of calling external contracts is that they can take over the control flow. In the reentrancy attack (a.k.a. recursive call attack), a malicious contract calls back into the calling contract before the first invocation of the function is finished. This may cause the different invocations of the function to interact in undesirable ways.

Remediation:

The best practices to avoid Reentrancy weaknesses are: Make sure all internal state changes are performed before the call is executed. This is known as the Checks-Effects-Interactions pattern Use a reentrancy lock.

References:

Ethereum Smart Contract Best Practices - Reentrancy

SWC-113 - DoS with Failed Call

CWE-703: Improper Check or Handling of Exceptional Conditions

Description:

External calls can fail accidentally or deliberately, which can cause a DoS condition in the contract. To minimize the damage caused by such failures, it is better to isolate each external call into its own transaction that can be initiated by the recipient of the call. This is especially relevant for payments, where it is better to let users withdraw funds rather than push funds to them automatically (this also reduces the chance of problems with the gas limit).

Remediation:

It is recommended to follow call best practices:Avoid combining multiple calls in a single transaction, especially when calls are executed as part of a loop. Always assume that external calls can fail. Implement the contract logic to handle failed calls

References:

ConsenSys Smart Contract Best Practices

SWC-116 - Block values as a proxy for time

CWE-829: Inclusion of Functionality from Untrusted Control Sphere

Description:

Contracts often need access to time values to perform certain types of functionality. Values such as block.timestamp, and block.number can give you a sense of the current time or a time delta, however, they are not safe to use for most purposes.

Remediation:

Developers should write smart contracts with the notion that block values are not precise, and the use of them can lead to unexpected effects. Alternatively, they may make use oracles..

References:

Safety: Timestamp dependence

Ethereum Smart Contract Best Practices - Timestamp Dependence

How do Ethereum mining nodes maintain a time consistent with the network?.

Solidity: Timestamp dependency, is it possible to do safely?.

Avoid using block.number as a timestamp

SWC Information Notes:

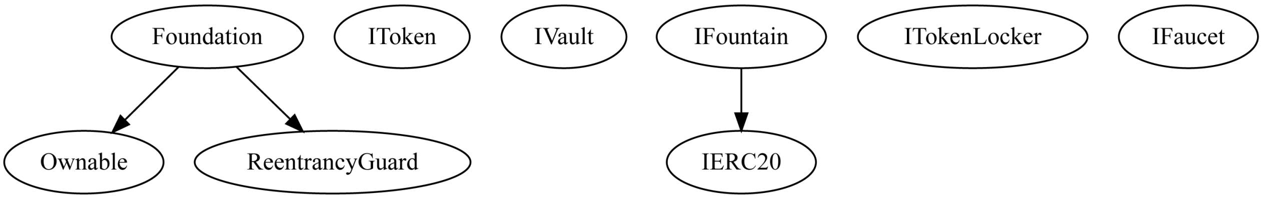
Auditor Notes:

Project Owner Notes:

Call Graph and Inheritance

The contract for Pinksale Subscription has the following call graph structure

The Project has a Total Supply of and has the following inheritance



Priviliged Functions (onlyOwner)

Function Name	Parameters	Visibility
renounceOwnership	none	public
transferOwnership	address newOwner	public
setPublicSaleStartTime		external
setWhitelistedUsers		external
_contribute		external

Important Notes To The Users:

- PinkSale is the number one Launchpad on the Crypto space, and they have the most trusted and dedicated team
- We had the opportunity to review their PinkLock02 and the contract is live and very secure.
- No high-risk Exploits/Vulnerabilities Were Found in the Source Code.
- We review the code and scan it for best practices, we have made suggestions to the team and they have addressed all of them.

Audit Passed

Social Media Checks

Social Media	URL	Result
Twitter	https://twitter.com/pinkecosystem	Pass
Medium		Pass
Website	https://www.pinksale.finance/subscription-pool/create?chain=BSC-Test	Pass
Telegram	https://t.me/pinkecosystem	Pass

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

Social Media Information Notes:

Auditor Notes: Reviewed the social media, customer could use some additional marketing. However everything is established

Project Owner Notes:

Disclaimer

CFGNINJA has conducted an independent audit to verify the integrity of and highlight any vulnerabilities or errors, intentional or unintentional, that may be present in the codes that were provided for the scope of this audit. This audit report does not constitute agreement, acceptance or advocation for the Project that was audited, and users relying on this audit 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 may be pursued by the Project in question, and that 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 completely 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 in this report. Information is provided 'as is', and CFGNINJA is under no covenant to the completeness, accuracy or solidity of the contracts audited. 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 and/or actions with regards to the information provided in this audit report.

The assessment services provided by CFGNINJA is subject to dependencies and under continuing development. You agree that your access and/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 and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives, false negatives, and other unpredictable results. The services may access, and depend upon, multiple layers of third-parties.