CFG NINJA AUDITS

Security Assessment

NFTPussies Token

May 26, 2022

Table of Contents

Audit Summary

This report has been prepared for NFTPussies 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.

Project Overview

Token Summary

Parameter	Result
Address	0x91D74E31A2734F4E519861aB9917048822f0B489
Name	NFTPussies
Token Tracker	NFTPussies (PUSSIES)
Decimals	18
Supply	1,000,000,000
Platform	Binance Smart Chain
compiler	v0.8.11+commit.d7f03943
Contract Name	NFTPussies
Optimization	Yes with 200 runs
LicenseType	MIT
Language	Solidity
Codebase	https://bscscan.com/token/0x91d74e31a2734f4e519861ab9917048822f0b489
Payment Tx	0xd9b38d4300acf731c8a394743053af9c15007bbd5052515040fb170f1c7e1c48

Project Overview

Risk Analysis Summary

Parameter	Result
Buy Tax	5%
Sale Tax	5%
Is honeypot?	Clean
Can edit tax?	Yes
Is anti whale?	No
Is blacklisted?	No
Is whitelisted?	No
Holders	Clean
Security Score	96/100
Auditor Score	96/100
Confidence Level	High

The following quick summary has been added to the project overview, however there are more details about the audit and their results please read every details.

Main Contract Assessed
Contract Name

Name	Contract	Live
NFTPussies	0x91D74E31A2734F4E519861aB9917048822f0B489	Yes

TestNet Contract Assessed
Contract Name

Name	Contract	Live
NFTPussies	0xcAF7C57fEA887a267bD4d5B66401E4e6453B06d1	Yes

Solidity Code Provided

SolID	File Sha-1	FileName
token	01cd0df67f1f6d7ea35d01d20b865137c42b0069	token.sol
token	cfe3ed784fcdabb3882972746147314de6c72fe7	IPresale.sol
token		

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

Mint Check

The Project Owners of NFTPussies does not have a mint function in the contract, owner cannot mint tokens after initial deploy

. .

The Project has a Total Supply of 1,000,000,000 and cannot mint any more than the Max Supply.

.

Mint Notes:

Auditor Notes:

Project Owner Notes:

Owner can't mint new coins

Fees Check

The Project Owners of NFTPussies does not have the ability to set fees higher than 25% .

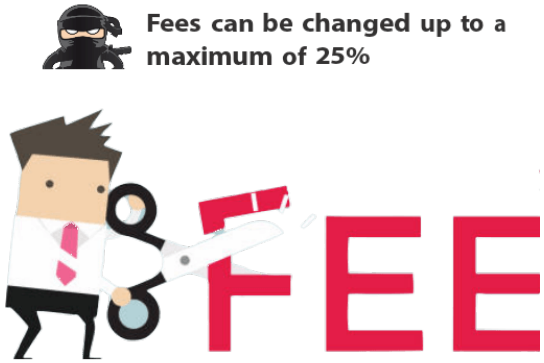
Team May have fees defined, however they dont have the ability to set those fees higher than 25%.

.

Tax Fee Notes:

Auditor Notes: Initial Tax 5% and cant be higher than 25%

Project Owner Notes: .



MaxTx Check

The Project Onwers of NFTPussies does not has the ability to set max tx amount

The Team allow any investors to swap, transfer or sale their total amount if needed.

Project Has No MaxTX

Pause Trade Check

The Project Owners of NFTPussies Owner can pause trading but he can move tokens.

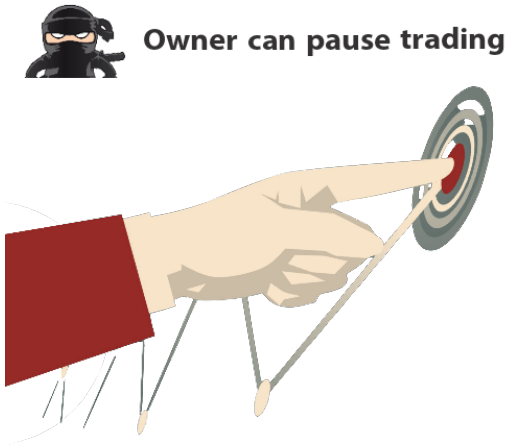
We recommend the team to only allow Open Trade and never use Stop Trade as this will be catastrofic for the project and investors.

We recommend the team to create a new contract without stop trade..

Pause Trade Notes:

Auditor Notes: Project Owner has the ability to pause trade

Project Owner Notes: My point of view would be that in case there is a security issue in the contract, vesting or generally anywhere in the platform which could lead to someone getting significant advantage we can pause trading and deal with it, so that we prevent loss of value/funds for the token.



Contract Ownership

The contract ownership of NFTPussies 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 0xecdff6dc6965cde3a80f79c8b7f13f47073a3435 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.

Liquidity Ownership

The token does not have liquidity at the moment of the audit, block 15164039

KYC Information

The Project Onwers of NFTPussies has provided KYC Documentation.

KYC Certificated can be found on the Following:
KYC Data

KYC Information Notes:

Auditor Notes: Asked project owner about KYC.

Project Owner Notes: Customer is KYC with PinkSale

Mythx Security Summary Checks

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

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

ID	Severity	Name	File	location
SWC-129	Pass	Typographical Error.	token.sol	L: 0 C: 0
SWC-130	Pass	Right-To-Left-Override control character (U +202E).	token.sol	L: 0 C: 0
SWC-131	Pass	Presence of unused variables.	token.sol	L: 0 C: 0
SWC-132	Pass	Unexpected Ether balance.	token.sol	L: 0 C: 0
SWC-133	Pass	Hash Collisions with Multiple Variable Length Arguments.	token.sol	L: 0 C: 0
SWC-134	Pass	Message call with hardcoded gas amount.	token.sol	L: 0 C: 0
SWC-135	Pass	Code With No Effects (Irrelevant/Dead Code).	token.sol	L: 0 C: 0
SWC-136	Pass	Unencrypted Private Data On-Chain.	token.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-110 - Assert Violation

CWE-670: Always-Incorrect Control Flow Implementation

Description:

The Solidity assert() function is meant to assert invariants. Properly functioning code should never reach a failing assert statement. A reachable assertion can mean one of two things:

A bug exists in the contract that allows it to enter an invalid state;
The assert statement is used incorrectly, e.g. to validate inputs.

Remediation:

Consider whether the condition checked in the assert() is actually an invariant. If not,

replace the assert() statement with a require() statement.If the exception is indeed caused by unexpected behaviour of the code, fix the underlying bug(s) that allow the assertion to be violated.

References:

The use of revert(), assert(), and require() in Solidity, and the new REVERT opcode in the EVM.

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-120 - Weak Sources of Randomness from Chain Attributes

CWE-330: Use of Insufficiently Random Values

Description:

Solidity allows for ambiguous naming of state variables when inheritance is used. Contract A with a variable x could inherit contract B that also has a state variable x defined. This would result in two separate versions of x, one of them being accessed from contract A and the other one from contract B. In more complex contract systems this condition could go unnoticed and subsequently lead to security issues.

Shadowing state variables can also occur within a single contract when there are multiple definitions on the contract and function level.

Remediation:

Using commitment scheme, e.g. RANDAO. Using external sources of randomness via oracles, e.g. Oraclize. Note that this approach requires trusting in oracle, thus it may be reasonable to use multiple oracles. Using Bitcoin block hashes, as they are more expensive to mine.

References:

How can I securely generate a random number in my smart contract?)

When can BLOCKHASH be safely used for a random number? When would it be unsafe?

The Run smart contract.

SWC Information Notes:

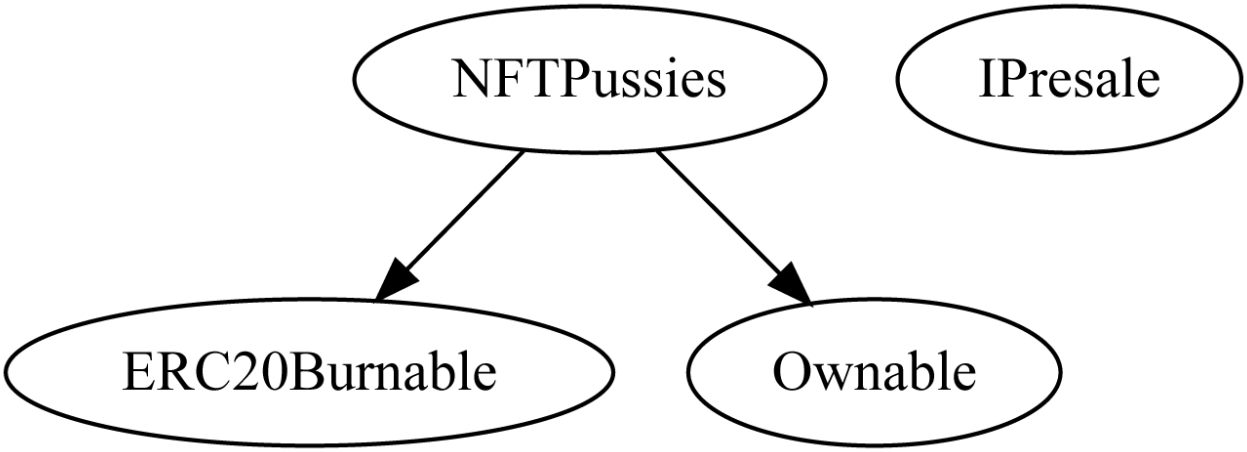
Auditor Notes:

Project Owner Notes:

Call Graph and Inheritance

The contract for NFTPussies has the following call graph structure

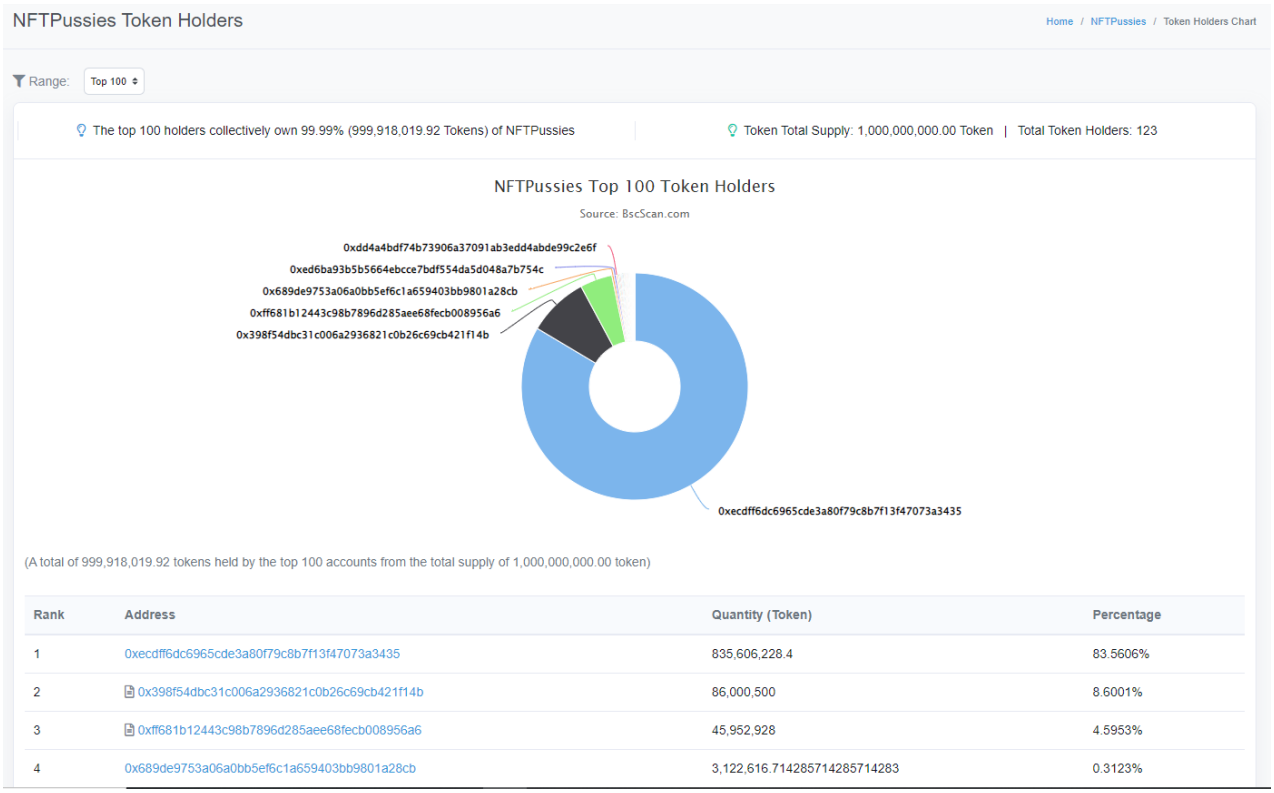
The Project has a Total Supply of 1,000,000,000 and has the following inheritance



Top Token Holders

The contract for NFTPussies has the following top token holders

.



Rank	Address	Quantity (Token)	Percentage
1	0xecdff6dc6965cde3a80f79c8b7f13f47073a3435	835,606,228.4	83.5606%
2	0x398f54dbc31c006a2936821c0b26c69cb421f14b	86,000,500	8.6001%
3	0xff681b12443c98b7896d285aee68fecb008956a6	45,952,928	4.5953%
4	0x689de9753a06a0bb5ef6c1a659403bb9801a28cb	3,122,616.714285714285714283	0.3123%

Priviliged Functions (onlyOwner)

Function Name	Parameters	Visibility
renounceOwnership	none	public
transferOwnership	address newOwner	public
setBuyTax	tax uint256	external
setSellTax	tax uint256	external
setAllowedContract	_contract address, enable bool	external
setLiquidityPair	_contract address, enable bool	external
setIgnoreTax	_contract address, enable bool	external
setTrading	enable bool	external
setAntibotBlocks	blocks uint256	external
rescueTokens	token address	external

Important Notes To The Users:

- NFT Pussies team is very responsive, we have asked the team to do several revisions of their contract and they have made those improvements.
- The team currently have a KYC with PinkSale.
- Owner can't charge fees up to 25%.
- Owner can't set max tx amount.
- Owner can pause trading.
- No high-risk Exploits/Vulnerabilities Were Found in the Source Code.

Audit Passed

Social Media Checks

Social Media	URL	Result
Twitter	https://twitter.com/PussiesNft	Pass
Reddit	https://www.instagram.com/nft_pussies/	Pass
Website	https://www.nftpussies.com/home	Pass
Telegram	https://t.me/NFT_Pussies_Community	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:

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.