



CFG NINJA AUDITS

Security Assessment

CreamFactory Token

October 6, 2023

Audit Status: Pass

Audit Edition: Advance














POWERED BY
BLADE POOL

Risk Analysis

















Classifications of Manual Risk Results

| Classification | Description |
|---|----------------------------------|
|  Critical | Danger or Potential Problems. |
|  Major | Be Careful or Fail test. |
|  Minor | Pass, Not-Detected or Safe Item. |
|  Informational | Function Detected |

Manual Code Review Risk Results

| Contract Priviledge | Description |
|--|--------------|
|  Buy Tax | 1% |
|  Sale Tax | 3% |
|  Cannot Sale | Pass |
|  Cannot Sale | Pass |
|  Max Tax | 3% |
|  Modify Tax | Pass |
|  Fee Check | Pass |
|  Is Honeypot? | Not Detected |
|  Trading Cooldown | Not Detected |
|  Can Pause Trade? | Pass |
|  Pause Transfer? | Not-Detected |



| Contract Priviledge | Description |
|--|--------------|
|  Max Tx? | Pass |
|  Is Anti Whale? | Not Detected |
|  Is Anti Bot? | Not Detected |
|  Is Blacklist? | Not Detected |
|  Blacklist Check | Pass |
|  is Whitelist? | Not Detected |
|  Can Mint? | Pass |
|  Is Proxy? | Not Detected |
|  Can Take Ownership? | Not Detected |
|  Hidden Owner? | Not Detected |
|  Owner | |
|  Self Destruct? | Not Detected |
|  Other? | Not Detected |
|  Other? | Not Detected |
|  Holders | 1 |
|  Auditor Confidence | Medium |

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.



Table of Contents

1 Assessment Summary

2 Project Overview

2.1 Token Summary

2.2 Risk Analysis Summary

2.3 Main Contract Assessed

3 Smart Contract Risk Checks

3.1 Mint Check

3.2 Fees Check

3.3 Blacklist Check

3.4 MaxTx Check

3.5 Pause Trade Check

3.6 Contract Ownership

3.7 Liquidity Ownership

3.8 KYC Check

4 Smart Contract Vulnerability Checks

4.1 Smart Contract Vulnerability Details

4.2 Smart Contract Inheritance Details

4.3 Smart Contract Privileged Functions

5 Technical Findings Details

6 Social Media Check(Informational)

7 Assessment Results and Notes(Important)

7.1 Score Results

8 Disclaimer



Assessment Summary

This report has been prepared for CreamFactory 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.



Project Overview

Token Summary

| Parameter | Result |
|---------------|---|
| Address | 0x7a29fb472f0e62bce6127f125e8d5ad2a9fa3839 |
| Name | CreamFactory |
| Token Tracker | CreamFactory (Cream-LP) |
| Decimals | 18 |
| Supply | 1,000,000,000 |
| Platform | Binance Smart Chain |
| compiler | v0.5.16+commit.9c3226ce |
| Contract Name | rebirth |
| Optimization | Yes with 200 runs |
| LicenseType | MIT |
| Language | Solidity |
| Codebase | https://bscscan.com/address/0x8cb25774298ff036A4849C5828b44F8D865e2B40#code |
| Payment Tx | Corporate |

MainNet Contract was Not Assessed

TestNet Contract Assessed



Contract Name

| Name | Contract | Live |
|--------------|--|------|
| CreamFactory | 0x57319f37434b252db0D09CDa3BF8FA2caD3166b1 | Yes |

Solidity Code Provided

| SolID | File Sha-1 | FileName |
|--------------|--|------------------|
| CreamFactory | c3f68e4bbd14b23bdbd26220a32dfc5765d86ae0 | CreamFactory.sol |
| CreamFactory | | |
| CreamFactory | | |
| CreamFactory | | |



Mint Check

The project owners of CreamFactory do 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:



Fees Check

The project owners of CreamFactory do not have the ability to set fees higher than 3% .

The team May have fees defined; however, they can't set those fees higher than 3% or may not be able to configure the same.

Tax Fee Notes:

Auditor Notes: Sala Tax is 0% and Buy Tax is 0%.

Project Owner Notes: Not Detected



Blacklist Check

The project owners of CreamFactory do not have a blacklist function their contract.

The Project allow owners to transfer their tokens without any restrictions.

Token owner cannot blacklist the contract: Malicious or compromised owners can trap contracts relying on tokens with a blacklist.

Blacklist Notes:

Auditor Notes: .

Project Owner Notes:



MaxTx Check

The Project Owners of CreamFactory cannot set max tx amount

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

MaxTX Notes:

Auditor Notes: Max tax is 3%.

Project Owner Notes:

Project Has No MaxTX



Pause Trade Check

The Project Owners of CreamFactory don't have the ability to stop or pause trading.

The Team has done a great job to avoid stop trading, and investors has the ability to trade at any given time without any problems

Pause Trade Notes:

Auditor Notes:

Project Owner Notes: .

Owner can't pause trading



Contract Ownership

The contract ownership of CreamFactory 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: [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

The token does not have liquidity at the moment of the audit, block 31987420

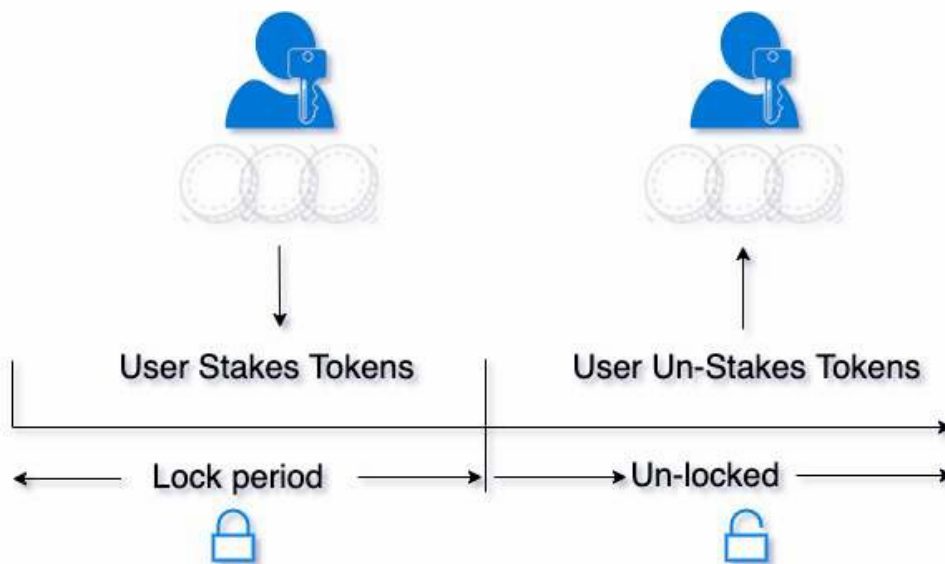
If liquidity is unlocked, then the token developers can do what is infamously known as 'rugpull'. Once investors start buying token from the exchange, the liquidity pool will accumulate more and more coins of established value (e.g., ETH or BNB or Tether). This is because investors are basically sending these tokens of value to the exchange, to get the new token. Developers can withdraw this liquidity from the exchange, cash in all the value and run off with it. Liquidity is locked by renouncing the ownership of liquidity pool (LP) tokens for a fixed time period, by sending them to a time-lock smart contract. Without ownership of LP tokens, developers cannot get liquidity pool funds back. This provides confidence to the investors that the token developers will not run away with the liquidity money. It is now a standard practice that all token developers follow, and this is what really differentiates a scam coin from a real one.

[Read More](#)



What is a Staking Contract

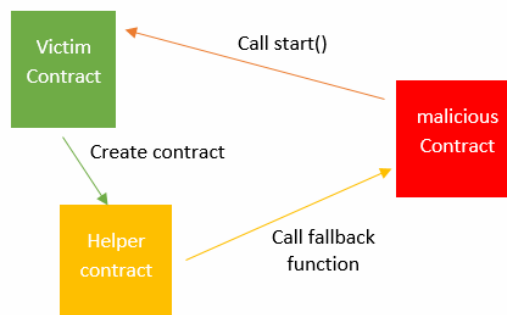
A smart contract which allows users to stake and un-stake a specified ERC20 token. Staked tokens are locked for a specific length of time (set by the contract owner at the outset). Once the time period has elapsed, the user can remove their tokens again.



The Project Owners of CreamFactory 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



KYC Information

The Project Owners of CreamFactory is not KYC.

KYC Information Notes:

Auditor Notes: KYC to be completed by PinkSale, project will be a SAFU Project.

Project Owner Notes:



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 | CreamFactory.sol | L: 0 C: 0 |
| SWC-101 | Pass | Integer Overflow and Underflow. | CreamFactory.sol | L: 0 C: 0 |
| SWC-102 | Pass | Outdated Compiler Version file. | CreamFactory.sol | L: 0 C: 0 |
| SWC-103 | Pass | A floating pragma is set. | CreamFactory.sol | L: 0 C: 0 |
| SWC-104 | Pass | Unchecked Call Return Value. | CreamFactory.sol | L: 0 C: 0 |
| SWC-105 | Pass | Unprotected Ether Withdrawal. | CreamFactory.sol | L: 0 C: 0 |
| SWC-106 | Pass | Unprotected SELFDESTRUCT Instruction | CreamFactory.sol | L: 0 C: 0 |
| SWC-107 | Pass | Read of persistent state following external call. | CreamFactory.sol | L: 0 C: 0 |
| SWC-108 | Low | State variable visibility is not set.. | CreamFactory.sol | L: 0 C: 0 |
| SWC-109 | Pass | Uninitialized Storage Pointer. | CreamFactory.sol | L: 0 C: 0 |
| SWC-110 | Pass | Assert Violation. | CreamFactory.sol | L: 0 C: 0 |



| ID | Severity | Name | File | location |
|---------|----------|--|------------------|-----------|
| SWC-111 | Pass | Use of Deprecated Solidity Functions. | CreamFactory.sol | L: 0 C: 0 |
| SWC-112 | Pass | Delegate Call to Untrusted Callee. | CreamFactory.sol | L: 0 C: 0 |
| SWC-113 | Pass | Multiple calls are executed in the same transaction. | CreamFactory.sol | L: 0 C: 0 |
| SWC-114 | Pass | Transaction Order Dependence. | CreamFactory.sol | L: 0 C: 0 |
| SWC-115 | Pass | Authorization through tx.origin. | CreamFactory.sol | L: 0 C: 0 |
| SWC-116 | Pass | A control flow decision is made based on The block.timestamp environment variable. | CreamFactory.sol | L: 0 C: 0 |
| SWC-117 | Pass | Signature Malleability. | CreamFactory.sol | L: 0 C: 0 |
| SWC-118 | Pass | Incorrect Constructor Name. | CreamFactory.sol | L: 0 C: 0 |
| SWC-119 | Pass | Shadowing State Variables. | CreamFactory.sol | L: 0 C: 0 |
| SWC-120 | Low | Potential use of block.number as source of randommness. | CreamFactory.sol | L: 0 C: 0 |
| SWC-121 | Pass | Missing Protection against Signature Replay Attacks. | CreamFactory.sol | L: 0 C: 0 |
| SWC-122 | Pass | Lack of Proper Signature Verification. | CreamFactory.sol | L: 0 C: 0 |
| SWC-123 | Pass | Requirement Violation. | CreamFactory.sol | L: 0 C: 0 |
| SWC-124 | Pass | Write to Arbitrary Storage Location. | CreamFactory.sol | L: 0 C: 0 |
| SWC-125 | Pass | Incorrect Inheritance Order. | CreamFactory.sol | L: 0 C: 0 |



| ID | Severity | Name | File | location |
|---------|----------|--|------------------|-----------|
| SWC-126 | Pass | Insufficient Gas Griefing. | CreamFactory.sol | L: 0 C: 0 |
| SWC-127 | Pass | Arbitrary Jump with Function Type Variable. | CreamFactory.sol | L: 0 C: 0 |
| SWC-128 | Pass | DoS With Block Gas Limit. | CreamFactory.sol | L: 0 C: 0 |
| SWC-129 | Pass | Typographical Error. | CreamFactory.sol | L: 0 C: 0 |
| SWC-130 | Pass | Right-To-Left-Override control character (U+202E). | CreamFactory.sol | L: 0 C: 0 |
| SWC-131 | Pass | Presence of unused variables. | CreamFactory.sol | L: 0 C: 0 |
| SWC-132 | Pass | Unexpected Ether balance. | CreamFactory.sol | L: 0 C: 0 |
| SWC-133 | Pass | Hash Collisions with Multiple Variable Length Arguments. | CreamFactory.sol | L: 0 C: 0 |
| SWC-134 | Pass | Message call with hardcoded gas amount. | CreamFactory.sol | L: 0 C: 0 |
| SWC-135 | Pass | Code With No Effects (Irrelevant/Dead Code). | CreamFactory.sol | L: 0 C: 0 |
| SWC-136 | Pass | Unencrypted Private Data On-Chain. | CreamFactory.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-108 - State Variable Default Visibility

CWE-710: Improper Adherence to Coding Standards

Description:

Labeling the visibility explicitly makes it easier to catch incorrect assumptions about who can access the variable.

Remediation:

Variables can be specified as being public, internal or private. Explicitly define visibility for all state variables.

References:

Ethereum Smart Contract Best Practices - Explicitly mark visibility in functions and state variables



Smart Contract Vulnerability Details

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?)

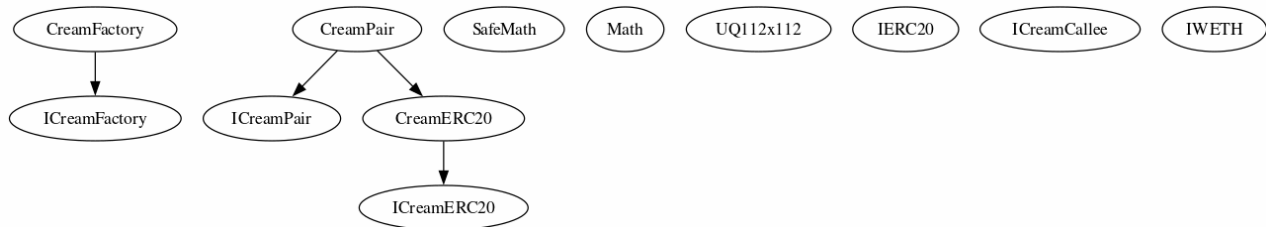
When can BLOCKHASH be safely used for a random number? When would it be unsafe?

The Run smart contract.



Inheritance

The contract for CreamFactory has the following inheritance structure.



Privileged Functions (onlyOwner)

Please Note if the contract is Renounced none of this functions can be executed.

| Function Name | Parameters | Visibility |
|---------------|------------|------------|
| | address | Public |



Smart Contract Advance Checks



| ID | Severity | Name | Result | Status |
|-------------|---------------|---|--------|-----------|
| Cream-LP-01 | Minor | Potential Sandwich Attacks. | Pass | Not-Found |
| Cream-LP-02 | Major | Function Visibility Optimization | Pass | Not-Found |
| Cream-LP-03 | Minor | Lack of Input Validation. | Fail | Pending |
| Cream-LP-04 | Major | Centralized Risk In addLiquidity. | Pass | Not-Found |
| Cream-LP-05 | Medium | Missing Event Emission. | Fail | Pending |
| Cream-LP-06 | Minor | Conformance with Solidity Naming Conventions. | Pass | Not-Found |
| Cream-LP-07 | Minor | State Variables could be Declared Constant. | Pass | Not-Found |
| Cream-LP-08 | Minor | Dead Code Elimination. | Pass | Not-Found |
| Cream-LP-09 | Major | Third Party Dependencies. | Pass | Not-Found |
| Cream-LP-10 | Major | Initial Token Distribution. | Pass | Not-Found |
| Cream-LP-11 | Major | Compiler Version is Outdated | Fail | Pending |
| Cream-LP-12 | Major | Centralization Risks In The X Role | Pass | Pending |
| Cream-LP-13 | Informational | Extra Gas Cost For User.. | Pass | Pending |



| ID | Severity | Name | Result | Status |
|-------------|---------------|--|--------|-----------|
| Cream-LP-14 | Major | Unnecessary Use Of SafeMath | Pass | Not-Found |
| Cream-LP-15 | Medium | Symbol Length Limitation due to Solidity Naming Standards. | Pass | Not-Found |
| Cream-LP-16 | Medium | Invalid collection of Taxes during Transfer. | Fail | Pending |
| Cream-LP-17 | Informational | Conformance to numeric notation best practice. | Pass | Not-Found |
| Cream-LP-18 | Major | Enable Trade and Exclude Exist to create a whitelist. | Pass | Not-Found |



Cream-LP-03 | Lack of Input Validation.

| Category | Severity | Location | Status |
|---------------|---|-----------------------------------|---|
| Volatile Code |  Minor | CreamFactory.sol: L: 588 C: 14 |  Pending |

Description

The given input is missing the check for the non-zero address.

The given input is missing the check for the .

Remediation



We advise the client to add the check for the passed-in values to prevent unexpected errors as below:

```
...  
    require(receiver != address(0), "Receiver is the zero address");  
...  
...  
    require(value X limitation, "Your not able to do this function");  
...
```

We also recommend customer to review the following function that is missing a required validation. .



Cream-LP-05 | Missing Event Emission.

| Category | Severity | Location | Status |
|---------------|--|--|---|
| Volatile Code |  Medium | CreamFactory.sol: L: 581 C: 14, L: 588 C: 14, L: 593 C: 14 |  Pending |

Description



Detected missing events for critical arithmetic parameters. There are functions that have no event emitted, so it is difficult to track off-chain changes. The linked code does not create an event for the transfer.

Remediation

Emit an event for critical parameter changes. It is recommended emitting events for the sensitive functions that are controlled by centralization roles.



Cream-LP-11 | Compiler Version is Outdated.

| Category | Severity | Location | Status |
|--------------|---|-----------------------------|---|
| Optimization |  Major | CreamFactory.sol: L: 0 C: 0 |  Pending |

Description

The current compiler is 0.5.16 and the latest compiler is 0.822, we recommend a N-2 in the compiler for better stability.


Remediation

Consider updating the code to the latest compiler version.

Project Action



Cream-LP-16 | Invalid collection of Taxes during Transfer.

| Category | Severity | Location | Status |
|---------------|----------|-----------------------------------|---|
| Logical Issue | Medium | CreamFactory.sol: L: 382 C: 14 |  Pending |

Description

The current taxes are collected during the transfer, however the current logic defined in the contract

```
uint256 _totalFees;  
if (_isExcludedFromFees[from] || _isExcludedFromFees[to] || swapping) {  
    _totalFees = 0;  
} else if (from == uniswapV2Pair) {  
    _totalFees = _totalFeesOnBuy;  
} else if (to == uniswapV2Pair) {  
    _totalFees = _totalFeesOnSell;  
} else {  
    _totalFees = 0;  
}  
  
if (_totalFees > 0) {  
    uint256 fees = (amount * _totalFees) / 100;  
    amount = amount - fees;  
    super._transfer(from, address(this), fees);  
}
```

due to the logic written in here may results in the contract not collecting the appropriate taxes for the project and may collect 0 taxes from it resulting in a major problem for the project.

Remediation

We advise revising the logic of taxes, to ensure taxes are collected and the swapAndLiquify is functioning. Since taxes are not collected, this other functions will never execute or get trigger.

Project Action








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 | 1 | 0 | 0 |
|  Medium | 0 | 0 | 0 |
|  Minor | 3 | 0 | 0 |
|  Informational | 0 | 0 | 0 |
| Total | 4 | 0 | 0 |



Social Media Checks

| Social Media | URL | Result |
|--------------|---|--------|
| Twitter | https://twitter.com/Cream_Swap | Pass |
| Other | | Fail |
| Website | https://creamswap.app/#/swap | Pass |
| Telegram | https://t.me/Cream_SWap | 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:



Assessment Results

Score Results

| Review | Score |
|---------------------|--------|
| Overall Score | 86/100 |
| Auditor Score | 85/100 |
| Review by Section | Score |
| Manual Scan Score | 35/53 |
| SWC Scan Score | 35 /37 |
| Advance Check Score | 16 /19 |

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 80 Points, if a project does not attain 80% is an automatic failure. Read our notes and final assessment below.

Audit Passed



Assessment Results

Important Notes:

- Multiple issues/vulnerabilities were found.
- This is a uniswap factory clone.
- <https://github.com/Uniswap/v2-core/blob/master/contracts/UniswapV2Factory.sol>
- The factory includes a code for pair creation, L: 286 C: 14
- Testing: <https://testnet.bscscan.com/address/0x39370372DDaC039b6582c42Ea06DA3b375ca8770#code>
- Contract by Jing.

Auditor Score =85

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 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.

