15 Dec 2022

DRIVENsecurity
Free Static Audit

Cryft
Static Code Analysis For Smart Contract

Request a complete audit

WWW.DRIVENECOSYSTEM.COM

Disclaimer

Accepting a project audit can be viewed as a sign of confidence and is typically the first indicator of trust for a project, but it does not guarantee that a team will not remove all liquidity, sell tokens, or engage in any other type of fraud. There is also no method to restrict private sale holders from selling their tokens. It is ultimately your obligation to read through all documentation, social media posts, and contract code for each particular project in order to draw your own conclusions and define your own risk tolerance.

DRIVENsecurity accepts no responsibility for any losses or encourages speculative investments. This audit's material is given solely for information reasons and should not be construed as investment advice.

Table Of Contents

Project Details

Name of the project:
Cryft

Type of the Smart Contract:
BEP-20

Chain:
Polygon

Address:
0xCd3fb58fCdaDff8DD7D23844dEA03f1e4B369500

Blockchain Explorer Link:
https://bscscan.com/

NOTE: This is a proxy smart contract, which means that its proxy can be changed at any time. The audited smart contract is the smart contract's most recently deployed proxy.

Static Analysis

SWC ISSUES	STATUS
Function Default Visibility	PASSED
Integer Overflow and Underflow	PASSED
Outdated Compiler Version	PASSED
Floating Pragma	LOW SEVERITY
Unchecked Call Return Value	PASSED
Unprotected Ether Withdrawal	PASSED
Unprotected SELFDESTRUCT Instruction	PASSED
Reentrancy	PASSED
State Variable Default Visibility	LOW SEVERITY
Uninitialized Storage Pointer	PASSED
Assert Violation	PASSED
Use of Deprecated Solidity Functions	PASSED
Delegatecall to Untrusted Callee	PASSED
DoS with Failed Call	PASSED
Transaction Order Dependence	PASSED
Authorization through tx.origin	PASSED
Block values as a proxy for time	PASSED
Signature Malleability	PASSED
Incorrect Constructor Name	PASSED
Shadowing State Variables	PASSED
Weak Sources of Randomness from Chain Attributes	PASSED
Missing Protection against Signature Replay Attacks	PASSED
Lack of Proper Signature Verification	PASSED
Requirement Violation	LOW SEVERITY

Static Analysis

SWC ISSUES	STATUS
Write to Arbitrary Storage Location	PASSED
Incorrect Inheritance Order	PASSED
Insufficient Gas Griefing	PASSED
Arbitrary Jump with Function Type Variable	PASSED
DoS With Block Gas Limit	PASSED
Typographical Error	PASSED
Right-To-Left-Override control character (U+202E)	PASSED
Presence of unused variables	PASSED
Unexpected Ether balance	PASSED
Hash Collisions With Multiple Variable Length Arguments	PASSED
Message call with hardcoded gas amount	PASSED
Code With No Effects	PASSED

Issues

ISSUE-01: A floating pragma is set.

File: INoBSDynamicReflector.sol
Location: L: 2 C: 0

The current pragma Solidity directive is ""^0.8.2"". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

ID: SWC-103

Severity: LOW

Issues

ISSUE-02: State variable visibility is not set.

File: INoBSDynamicReflector.sol
Location: L: 13 C: 24896

It is best practice to set the visibility of state variables explicitly. The default visibility for "isProcessExcluded" is internal. Other possible visibility settings are public and private.

ID: SWC-108

Severity: LOW

Issues

ISSUE-03: Requirement violation.

File: ContextUpgradeable.sol
Location: L: 38 C: 2410

A requirement was violated in a nested call and the call was reverted as a result. Make sure valid inputs are provided to the nested call (for instance, via passed arguments).

ID: SWC-123

Severity: LOW

Issues

ISSUE-04: Requirement violation.

File: INoBSDynamicReflector.sol
Location: L: 13 C: 31

A requirement was violated in a nested call and the call was reverted as a result. Make sure valid inputs are provided to the nested call (for instance, via passed arguments).

ID: SWC-123

Severity: LOW

Conclusion

Passing a static code audit does not imply that the smart contract is safe and free of backdoors or malicious code written by the developer.

A successful static code audit verifies that the smart contract was created using proper Solidity principles and that the smart contract is "deployable."

DRIVENecosystem team did not manually examine the code or run any manual tests on this specific smart contract.

Free Audits vs Paid Audits

Free Static Audits are based on a static code analysis, which is a brief syntax check.

Manual code verifications, testing on test networks, testing on mainnet (optional), static analysis, and penetration testing are all part of the paid technical audits.

Our customers can choose from a variety of audits. We have included a list of our items and their prices below.

- Technical Audit: starting from $2,500
- Fundamental Audit: $2,800
- Wallet Forensics: $900
- KYC Verification: $450

Request a complete audit:
www.drivenecosystem.com/drivensecurity

15 Dec 2022

Thank you!

Request a complete audit
www.drivenecosystem.com