

16 Dec 2022

DRIVENsecurity

Free Static Audit

Encyclopedia wTa

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	3
Static Analysis	4
Issues	6
Conclusion	9
Free Audits vs Paid Audits	10



Project Details

Name of the project:

Encyclopedia wTa

Type of the Smart Contract:

BEP-20

Chain:

BNB Smart Chain

Address:

0xEa4bCFFf56Ef4aeDd7f41Fc260F13f2eEE7DeFCe

Blockchain Explorer Link:

<https://bscscan.com/>



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Outdated Compiler Version	PASSED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	MEDIUM SEVERITY
Unprotected Ether Withdrawal	PASSED
Unprotected SELFDESTRUCT Instruction	PASSED
Reentrancy	PASSED
State Variable Default Visibility	PASSED
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	PASSED



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
Authorization through tx.origin	LOW SEVERITY



Issues

ISSUE-01: A floating pragma is set.

File: CoinToken.sol

Location: L: 8 C: 0

The current pragma Solidity directive is " "`^0.8.9`" ". 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: Unchecked return value from low-level external call.

File: CoinToken.sol

Low-level external calls return a boolean value. If the callee halts with an exception, 'false' is returned and execution continues in the caller. The caller should check whether an exception happened and react accordingly to avoid unexpected behaviour. For example it is often desirable to wrap low-level external calls in require() so the transaction is reverted if the call fails.

Location:

1 L: 978 C: 20
2 L: 979 C: 20
3 L: 980 C: 20
4 L: 983 C: 24

ID: SWC-104

Severity: **MEDIUM**



Issues

ISSUE-03: Use of "tx.origin" as a part of authorization control.

File: CoinToken.sol

Location: L: 1008 C: 31

Using "tx.origin" as a security control can lead to authorization bypass vulnerabilities. Consider using "msg.sender" unless you really know what you are doing.

ID: SWC-115

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

16 Dec 2022

Thank you!

Request a complete audit
www.drivenecosystem.com