*SECURITY AUDIT OF*

# UNS TOKEN SMART CONTRACT

**Public Report**

*Dec 22, 2022*

**Verichains Lab**

info@verichains.io

https://www.verichains.io

*Driving Technology > Forward*

# ABBREVIATIONS

| Name | Description |
|---|---|
| **Ethereum** | An open source platform based on blockchain technology to create and distribute smart contracts and decentralized applications. |
| **Ether (ETH)** | A cryptocurrency whose blockchain is generated by the Ethereum platform. Ether is used for payment of transactions and computing services in the Ethereum network. |
| **Smart contract** | A computer protocol intended to digitally facilitate, verify or enforce the negotiation or performance of a contract. |
| **Solidity** | A contract-oriented, high-level language for implementing smart contracts for the Ethereum platform. |
| **Solc** | A compiler for Solidity. |
| **ERC20** | ERC20 (BEP20 in Binance Smart Chain or *x*RP20 in other chains) tokens are blockchain-based assets that have value and can be sent and received. The primary difference with the primary coin is that instead of running on their own blockchain, ERC20 tokens are issued on a network that supports smart contracts such as Ethereum or Binance Smart Chain. |

# EXECUTIVE SUMMARY

This Security Audit Report was prepared by Verichains Lab on Dec 22, 2022. We would like to thank the UNS for trusting Verichains Lab in auditing smart contracts. Delivering high-quality audits is always our top priority.

This audit focused on identifying security flaws in code and the design of the UNS Token Smart Contract. The scope of the audit is limited to the source code files provided to Verichains. Verichains Lab completed the assessment using manual, static, and dynamic analysis techniques.

During the audit process, the audit team had identified **NO VULNERABILITY** issue in the contract code.

# TABLE OF CONTENTS

# 1. MANAGEMENT SUMMARY

## 1.1. About UNS Token Smart Contract

UNS token is a BEP-20 token with a vision to become a multisector utility token. As a first step towards building its ecosystem, UNS token will have its own Cryptocurrency Exchange called as UNS Exchange, in which UNS token will be the governing native token followed by partnering with various projects in 4 different sectors of Education, Healthcare, Agriculture and Energy.

## 1.2. Audit scope

This audit focused on identifying security flaws in code and the design of the UNS Token Smart Contract.

The audited contract is the UNS Token Smart Contract that deployed on Binance Smart Chain Mainnet at address `0xecf99ab23c11ddc6520252105308c251001aefbb`. The details of the deployed smart contract are listed in Table 1.

| FIELD | VALUE |
|---|---|
| **Contract Name** | UnsToken |
| **Contract Address** | 0xecf99ab23c11ddc6520252105308c251001aefbb |
| **Compiler Version** | v0.8.7+commit.e28d00a7 |
| **Optimization Enabled** | No with 200 runs |
| **Explorer** | *https://bscscan.com/address/0xecf99ab23c11ddc6520252105308c251001aefbb* |

*Table 1. The deployed smart contract details*

## 1.3. Audit methodology

Our security audit process for smart contract includes two steps:

- Smart contract codes are scanned/tested for commonly known and more specific vulnerabilities using public and RK87, our in-house smart contract security analysis tool.

- Manual audit of the codes for security issues. The contracts are manually analyzed to look for any potential problems.

Following is the list of commonly known vulnerabilities that were considered during the audit of the smart contract:

- Integer Overflow and Underflow
- Timestamp Dependence
- Race Conditions
- Transaction-Ordering Dependence
- DoS with (Unexpected) revert
- DoS with Block Gas Limit
- Gas Usage, Gas Limit and Loops
- Redundant fallback function
- Unsafe type Inference
- Reentrancy
- Explicit visibility of functions state variables (external, internal, private and public)
- Logic Flaws

For vulnerabilities, we categorize the findings into categories as listed in table below, depending on their severity level:

| SEVERITY LEVEL | DESCRIPTION |
|---|---|
| **CRITICAL** | A vulnerability that can disrupt the contract functioning; creates a critical risk to the contract; required to be fixed immediately. |
| **HIGH** | A vulnerability that could affect the desired outcome of executing the contract with high impact; needs to be fixed with high priority. |
| **MEDIUM** | A vulnerability that could affect the desired outcome of executing the contract with medium impact in a specific scenario; needs to be fixed. |
| **LOW** | An issue that does not have a significant impact, can be considered as less important. |

*Table 2. Severity levels*

## 1.4. Disclaimer

Please note that security auditing cannot uncover all existing vulnerabilities, and even an audit in which **NO VULNERABILITIES** are found is not a guarantee for a 100% secure smart contract. However, auditing allows discovering vulnerabilities that were unobserved, overlooked during development and areas where additional security measures are necessary.

# 2. AUDIT RESULT

## 2.1. Overview

The UNS Token Smart Contract was written in Solidity language, with the required version to be ^0.8.7. The source code was written based on OpenZeppelin's library.

### 2.1.1. Token contract

UNS Token Smart Contract extends ERC20, ERC20Burnable and AccessControl contracts. AccessControl allows the contract to implement role-based access control mechanisms. There are 2 roles: DEFAULT_ADMIN_ROLE and BLACKLIST_ROLE. At deployment, an amount of tokens equal to totalSupply will be minted for the contract deployer. The DEFAULT_ADMIN_ROLE role will be given to the contract deployer, who can then assign any roles to anyone. Transferring tokens is not permitted for users in the BLACKLIST_ROLE role. ERC20Burnable allows token holders to destroy both their own tokens and those that they have an allowance for.

Any token in the contract can be transferred to their wallet by a user with the DEFAULT_ADMIN_ROLE role.

Table below lists some properties of the audited PiBridge token contract (as of the report writing time).

| PROPERTY | VALUE |
|---|---|
| **Name** | UNS Token |
| **Symbol** | UNS |
| **Decimals** | 18 |
| **Total Supply** | 1,000,000,000 (x$10^{18}$)<br>Note: the number of decimals is 18, so the total representation token will be 1,000,000,000 or 1 billion. |

*Table 3. UNS Token properties*

## 2.2. Findings

During the audit process, the audit team found **NO VULNERABILITY** in the given version of UNS Token Smart Contract.

# APPENDIX

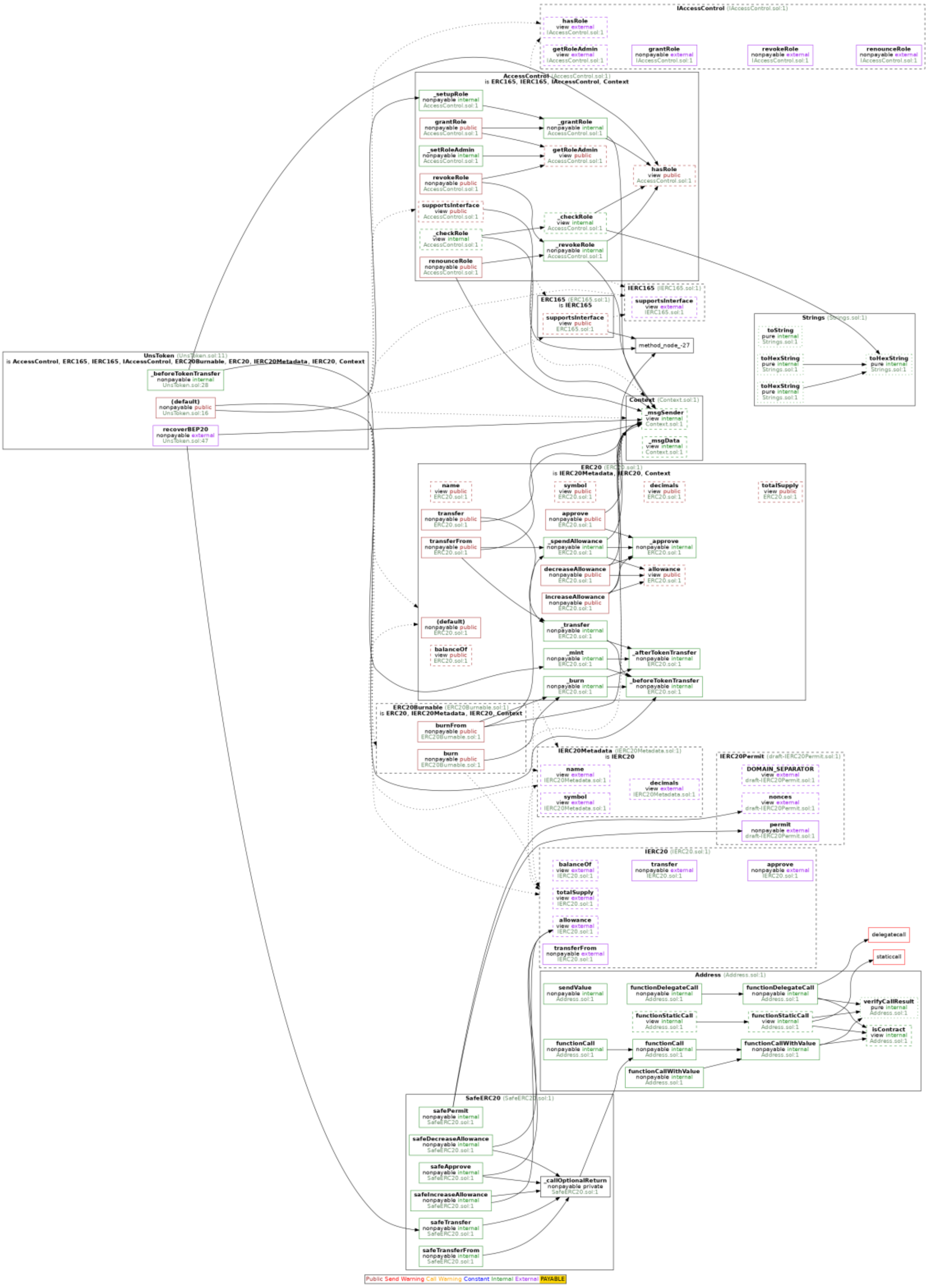
IAccessControl (IAccessControl.sol:1)
hasRole view external IAccessControl.sol:1
getRoleAdmin view external IAccessControl.sol:1
grantRole nonpayable external IAccessControl.sol:1
revokeRole nonpayable external IAccessControl.sol:1
renounceRole nonpayable external IAccessControl.sol:1
AccessControl (AccessControl.sol:1) is ERC165, IERC165, IAccessControl, Context
_setupRole nonpayable internal AccessControl.sol:1
grantRole nonpayable public AccessControl.sol:1
_setRoleAdmin nonpayable internal AccessControl.sol:1
revokeRole nonpayable public AccessControl.sol:1
supportsInterface view public AccessControl.sol:1
_checkRole view internal AccessControl.sol:1
renounceRole nonpayable public AccessControl.sol:1
_grantRole nonpayable internal AccessControl.sol:1
getRoleAdmin view public AccessControl.sol:1
_checkRole view internal AccessControl.sol:1
_revokeRole nonpayable internal AccessControl.sol:1
hasRole view public AccessControl.sol:1
IERC165 (IERC165.sol:1)
supportsInterface view external IERC165.sol:1
ERC165 (ERC165.sol:1) is IERC165
supportsInterface view public ERC165.sol:1
method_node_-27
Strings (Strings.sol:1)
toString pure internal Strings.sol:1
toHexString pure internal Strings.sol:1
toHexString pure internal Strings.sol:1
toHexString pure internal Strings.sol:1
UnsToken (UnsToken.sol:11)
is AccessControl, ERC165, IERC165, IAccessControl, ERC20Burnable, ERC20, IERC20Metadata, IERC20, Context
_beforeTokenTransfer nonpayable internal UnsToken.sol:28
(default) nonpayable public UnsToken.sol:16
recoverBEP20 nonpayable external UnsToken.sol:47
Context (Context.sol:1)
_msgSender view internal Context.sol:1
_msgData view internal Context.sol:1
ERC20 (ERC20.sol:1) is IERC20Metadata, IERC20, Context
name view public ERC20.sol:1
symbol view public ERC20.sol:1
decimals view public ERC20.sol:1
totalSupply view public ERC20.sol:1
transfer nonpayable public ERC20.sol:1
approve nonpayable public ERC20.sol:1
transferFrom nonpayable public ERC20.sol:1
_spendAllowance nonpayable internal ERC20.sol:1
_approve nonpayable internal ERC20.sol:1
decreaseAllowance nonpayable public ERC20.sol:1
allowance view public ERC20.sol:1
increaseAllowance nonpayable public ERC20.sol:1
(default) nonpayable public ERC20.sol:1
_transfer nonpayable internal ERC20.sol:1
balanceOf view public ERC20.sol:1
_mint nonpayable internal ERC20.sol:1
_afterTokenTransfer nonpayable internal ERC20.sol:1
_burn nonpayable internal ERC20.sol:1
_beforeTokenTransfer nonpayable internal ERC20.sol:1
ERC20Burnable (ERC20Burnable.sol:1) is ERC20, IERC20Metadata, IERC20, Context
burnFrom nonpayable public ERC20Burnable.sol:1
burn nonpayable public ERC20Burnable.sol:1
IERC20Metadata (IERC20Metadata.sol:1) is IERC20
name view external IERC20Metadata.sol:1
symbol view external IERC20Metadata.sol:1
decimals view external IERC20Metadata.sol:1
IERC20Permit (draft-IERC20Permit.sol:1)
DOMAIN_SEPARATOR view external draft-IERC20Permit.sol:1
nonces view external draft-IERC20Permit.sol:1
permit nonpayable external draft-IERC20Permit.sol:1
IERC20 (IERC20.sol:1)
balanceOf view external IERC20.sol:1
transfer nonpayable external IERC20.sol:1
approve nonpayable external IERC20.sol:1
totalSupply view external IERC20.sol:1
allowance view external IERC20.sol:1
transferFrom nonpayable external IERC20.sol:1
delegatecall
staticcall
Address (Address.sol:1)
sendValue nonpayable internal Address.sol:1
functionDelegateCall nonpayable internal Address.sol:1
functionDelegateCall nonpayable internal Address.sol:1
verifyCallResult pure internal Address.sol:1
functionStaticCall view internal Address.sol:1
functionStaticCall view internal Address.sol:1
isContract view internal Address.sol:1
functionCall nonpayable internal Address.sol:1
functionCall nonpayable internal Address.sol:1
functionCallWithValue nonpayable internal Address.sol:1
functionCallWithValue nonpayable internal Address.sol:1
SafeERC20 (SafeERC20.sol:1)
safePermit nonpayable internal SafeERC20.sol:1
safeDecreaseAllowance nonpayable internal SafeERC20.sol:1
safeApprove nonpayable internal SafeERC20.sol:1
safeIncreaseAllowance nonpayable internal SafeERC20.sol:1
safeTransfer nonpayable internal SafeERC20.sol:1
safeTransferFrom nonpayable internal SafeERC20.sol:1
_callOptionalReturn nonpayable private SafeERC20.sol:1
Public Send Warning Call Warning Constant Internal External PAYABLE

*Image 1. UNS Token Smart Contract call graph*

# 3. VERSION HISTORY

| Version | Date | Status/Change | Created by |
|---|---|---|---|
| **1.0** | *Dec 22, 2022* | Public Report | Verichains Lab |

*Table 4. Report versions history*