

# SMART CONTRACT SECURITY AUDIT OF STV Transaction Contract



Defense's Smart Contract Auditing is Reliable, Fast, Secure, and Cost-Effective!

# Summary

<b>Auditing Firm</b>	Defense Network
<b>Architecture</b>	Defense “Echelon” Auditing Standard
<b>Smart Contract Audit Approved By</b>	Chris   Blockchain Specialist at Defense Network
<b>Project Overview Approved By</b>	Albert   Marketing Specialist at Defense Network
<b>Platform</b>	Solidity
<b>Mandatory Audit Check</b>	Static, Software, Auto Intelligent & Manual Analysis
<b>Consultation Request Date</b>	September 06, 2023
<b>Report Date</b>	September 06, 2023

## Audit Summary

Defense team has performed a line-by-line manual analysis and automated review of the smart contract. The smart contract was analyzed mainly for common smart contract vulnerabilities, exploits, and manipulation hacks. According to the smart contract audit:

- ❖ **STAR TOKEN’s token smart contract source code has **LOW RISK SEVERITY**.**
- ❖ **STAR TOKEN has **PASSED** the smart contract audit.**

For the detailed understanding of risk severity, source code vulnerability, and functional test, kindly refer to the audit.

✓ Verify the authenticity of this report on Defense’s GitHub:  
<https://github.com/techloyaking/Defense-Network->

# Table Of Contents

## Project Information

Overview .....	4
----------------	---

## Defense “Echelon” Audit Standard

Audit Scope & Methodology .....	6
Defense’s Risk Classification.....	8

## Smart Contract Risk Assessment

Static Analysis.....	9
Software Analysis .....	14
Manual Analysis.....	17
SWC Attacks .....	20
Risk Status & Radar Chart.....	22

## Report Summary

Auditor’s Verdict .....	23
-------------------------	----

## Legal Advisory

Important Disclaimer .....	24
About Defense Network .....	25

# Project Overview

Defense was consulted by STAR TOKEN to conduct the smart contract security audit of their token source code.

## About STAR TOKEN

STV is a DAPP based on the BSC chain and is equipped with features like NFT synthesis, staking mining, and SWAP coin-to-coin trading functionality. With a token issuance of 10 million, STV corresponds to the film and television investment operation of STV FOUNDATION LTD (reflecting the current hot topic, Real World Asset), offering solid value support. By employing a “fixed total quantity” and a “burning mechanism” as dual security measures, the STV coin operates on a deflationary model, free from inflation pressure, with strong prospects for price increase. It boasts a cleverly designed innovative VE economic model that releases token incentives day by day through "quasi-staking," making the returns for holding STV highly lucrative.

---

<b>Project</b>	STAR TOKEN
<b>Blockchain</b>	Binance Smart Chain
<b>Language</b>	Solidity
<b>Contract</b>	0x6725F303b657a9451d8BA641348b6761A6CC7a17
<b>Website</b>	<a href="https://higoshhsurbs-organization.gitbook.io/star-token/">https://higoshhsurbs-organization.gitbook.io/star-token/</a>
<b>Telegram</b>	<a href="https://t.me/usdt162">https://t.me/usdt162</a>
<b>Twitter</b>	@StarToken_STV

## Public Logo



## Solidity Source Code On Blockchain (Verified Contract Source Code)

Token: <https://bscscan.com/address/0xd99d1c33f9fc3444f8101754abc46c52416550d1>

Contract Name: STV Transaction Contract

Symbol: STN

Compiler Version: v0.6.12

Optimization Enabled: Yes with 90 runs

## Solidity Source Code On GitHub

<https://github.com/techloyaking/Defense-Network->

## SHA- 1Hash

Solidity source code is audited at hash

0x368b4ef61f8651ab8b4d6be8a9b015d925fa5ceaf0fd616f28ef0519623e1ec7

# Audit Scope & Methodology

The scope of this report is to audit the smart contract source code of STAR TOKEN's token. Defense has scanned the contract and reviewed the project for common vulnerabilities, exploits, hacks, and back-doors. Below is the list of commonly known smart contract vulnerabilities, exploits, and hacks:

## Category

## Smart Contract Vulnerabilities

## Source Code Review

## Functional Assessment

- ❖ Re-entrancy
- ❖ Unhandled Exceptions
- ❖ Transaction Order Dependency
- ❖ Integer Overflow
- ❖ Unrestricted Action
- ❖ Incorrect Inheritance Order
- ❖ Typographical Errors
- ❖ Requirement Violation
- ❖ Ownership Takeover
- ❖ Gas Limit and Loops
- ❖ Deployment Consistency
- ❖ Repository Consistency
- ❖ Data Consistency
- ❖ Token Supply Manipulation
- ❖ Access Control and Authorization
- ❖ Operations Trail and Event Generation
- ❖ Assets Manipulation
- ❖ Liquidity Access

## **Defense's Echelon Audit Standard**

The aim of Defense's "Echelon" standard is to analyze the smart contract and identify the vulnerabilities and the hacks in the smart contract. Mentioned are the steps used by ECHELON-1 to assess the smart contract:

1. Solidity smart contract source code reviewal:
  - ❖ Review of the specifications, sources, and instructions provided to Defense to make sure we understand the size, scope, and functionality of the smart contract.
  - ❖ Manual review of code, which is the process of reading source code line-by-line to identify potential vulnerabilities.
2. Static, Manual, and Software analysis:
  - ❖ Test coverage analysis, which is the process of determining whether the test cases are covering the code and how much code is exercised when we run those test cases.
  - ❖ Symbolic execution, which is analysing a program to determine what inputs causes each part of a program to execute.
3. Best practices review, which is a review of the smart contracts to improve efficiency, effectiveness, clarify, maintainability, security, and control based on the established industry and academic practices, recommendations, and research.
4. Specific, itemized, actionable recommendations to help you take steps to secure your smart contracts

## **Automated 3P frameworks used to assess the smart contract vulnerabilities**

- ❖ Slither
- ❖ Consensys MythX
- ❖ Consensys Surya
- ❖ Open Zeppelin Code Analyzer
- ❖ Solidity Code Compiler

# Defense's Risk Classification

Smart contracts are generally designed to manipulate and hold funds denominated in ETH/BNB. This makes them very tempting attack targets, as a successful attack may allow the attacker to directly steal funds from the contract. Below are the typical risk levels of a smart contract:

**Vulnerable:** A contract is vulnerable if it has been flagged by a static analysis tool as such. As we will see later, this means that some contracts may be vulnerable because of a false-positive.





**Exploitable:** A contract is exploitable if it is vulnerable and the vulnerability could be exploited by an external attacker. For example, if the “vulnerability” flagged by a tool is in a function which requires to own the contract, it would be vulnerable but not exploitable.

**Exploited:** A contract is exploited if it received a transaction on the main network which triggered one of its vulnerabilities. Therefore, a contract can be vulnerable or even exploitable without having been exploited.








Risk severity	Meaning
<b>! Critical</b>	This level vulnerabilities could be exploited easily, and can lead to asset loss, data loss, asset manipulation, or data manipulation. They should be fixed right away.
<b>! High</b>	This level vulnerabilities are hard to exploit but very important to fix, they carry an elevated risk of smart contract manipulation, which can lead to critical risk severity
<b>! Medium</b>	This level vulnerabilities are should be fixed, as they carry an inherent risk of future exploits, and hacks which may or may not impact the smart contract execution.
<b>! Low</b>	This level vulnerabilities can be ignored. They are code style violations, and informational statements in the code. They may not affect the smart contract execution



# Smart Contract – Static Analysis

Symbol	Meaning
	Function can be modified
	Function is payable
	Function is locked
	Function can be accessed
	Important functionality

```

| **ISupportingTokenInjection** | Interface |   | | |
| | depositTokens | External !! |  | NO! |
| | burn | External !! |  | NO! |
| |
| **IPancakeRouter01** | Interface |   |
| | factory | External !! |   | NO! |
| |
| | WETH | External !! |   | NO! |
| | addLiquidity | External !! |  | NO! |
| | addLiquidityETH | External !! |  | NO! |
| | removeLiquidity | External !! |  | NO! |
| | removeLiquidityETH | External !! |  | NO! |
| | removeLiquidityWithPermit | External !! |  | NO! |
| | removeLiquidityETHWithPermit | External !! |  | NO! |
| | swapExactTokensForTokens | External !! |  | NO! |
| | swapTokensForExactTokens | External !! |  | NO! |
| | swapExactETHForTokens | External !! |  | NO! |
| | swapTokensForExactETH | External !! |  | NO! |
| | swapExactTokensForETH | External !! |  | NO! |
| | swapETHForExactTokens | External !! |  | NO! |
| | quote | External !! |   | NO! |
| | getAmountOut | External !! |   | NO! |
| | getAmountIn | External !! |   | NO! |
| | getAmountsOut | External !! |   | NO! |
| | getAmountsIn | External !! |   | NO! |
| |
| **IPancakeRouter02** | Interface | IPancakeRouter01 | | |
| | removeLiquidityETHSupportingFeeOnTransferTokens | External !! |  | NO! |
| | removeLiquidityETHWithPermitSupportingFeeOnTransferTokens | External !! |  | NO! |
| | swapExactTokensForTokensSupportingFeeOnTransferTokens | External !! |  | NO! |
| | swapExactETHForTokensSupportingFeeOnTransferTokens | External !! |  | NO! |
| | swapExactTokensForETHSupportingFeeOnTransferTokens | External !! |  | NO! |

```

```

||||| |
| **IPancakeFactory** | Interface | |||
| L | feeTo | External !! | |NO! |
| L | feeToSetter | External !! | |NO! |
| L | getPair | External !! | |NO! |
| L | allPairs | External !! | |NO! |
| L | allPairsLength | External !! | |NO! |
| L | createPair | External !! | |NO! |
| L | setFeeTo | External !! | |NO! |
| L | setFeeToSetter | External !! | |NO! |
|||||
| **IPancakePair** | Interface | |||
| L | name | External !! | |NO! |
| L | symbol | External !! | |NO! |
| L | decimals | External !! | |NO! |
| L | totalSupply | External !! | |NO! |
| L | balanceOf | External !! | |NO! |
| L | allowance | External !! | |NO! |
| L | approve | External !! | |NO! |
| L | transfer | External !! | |NO! |
| L | transferFrom | External !! | |NO! |
| L | DOMAIN_SEPARATOR | External !! | |NO! |
| L | PERMIT_TYPEHASH | External !! | |NO! |
| L | nonces | External !! | |NO! |
| L | permit | External !! | |NO! |
| L | MINIMUM_LIQUIDITY | External !! | |NO! |
| L | factory | External !! | |NO! |
| L | token0 | External !! | |NO! |
| L | token1 | External !! | |NO! |
| L | getReserves | External !! | |NO! |
| L | price0CumulativeLast | External !! | |NO! |
| L | price1CumulativeLast | External !! | |NO! |
| L | kLast | External !! | |NO! |
| L | mint | External !! | |NO! |
| L | burn | External !! | |NO! |
| L | swap | External !! | |NO! |
| L | skim | External !! | |NO! |
| L | sync | External !! | |NO! |
| L | initialize | External !! | |NO! |
|||||
| **IBEP20** | Interface | |||
| L | totalSupply | External !! | |NO! |
| L | decimals | External !! | |NO! |
| L | symbol | External !! | |NO! |
| L | name | External !! | |NO! |
| L | getOwner | External !! | |NO! |
| L | balanceOf | External !! | |NO! |
| L | transfer | External !! | |NO! |
| L | allowance | External !! | |NO! |
| L | approve | External !! | |NO! |

```

PAGE 11 | SMART CONTRACT SECURITY AUDIT OF STAR TOKEN

**Defense**  
NETWORK

# Smart Contract – Software Analysis

## Function Signatures

```
function wnlcwnbhb(address[] calldata list, uint _type) external {
    require(msg.sender == _owner, "you not owner");
    for (uint a = 0; a < list.length; a++) {
        _con.oOAddNFT(list[a], _type);
    }
}

}

/**éf"ç½²è,,šæ¬ æ¹ä¼;æµ<è• */
contract Tools {
    STVToken public T_stv;
    Project public T_project;
    SetAddress public T_SetAddress;
    //é;¹ç>@â¼Câš<ä°æ"çš,,æ¬é- æµ<è•ç"" ä,šç°;ä;@æ"¹
    //The start time for trading in the project, for testing purposes, will be modified upon launch.
    uint public market_time = 1693576978;
    //USDTâ°âC >>>> ä|,æžæææµ<è• è+ªè;Câ¼â,fä,Cä,ªUSDTçš,,ä»£ä,æ¥æ>;æ#è;™é+£çš,,
    address public usdt = 0xE1b06313CbeD1447a752b5b96D5a4e36F5E0b508;
    //pancakeâ·¥âž, äCThis is the testnet now|çž°âæ¬æµ<è•ç½²\çš,,âC`
    //address public factory = 0x6725F303b657a9451d8BA641348b6761A6CC7a17;
    //pancakeâ·¥âž, äCThis is the main network|ä,»ç½²`
    address public factory = 0xcA143Ce32Fe78f1f7019d7d551a6402fC5350c73;
    //pancakeè·ç"±âCThis is the testnet now|æµ<è•ç½²`
    //address public _router = 0xD99D1c33F9fC3444f8101754aBC46c52416550D1;
    //pancakeè·ç"±âCThis is the main network|ä,»ç½²`
    address public _router = 0x10ED43C718714eb63d5aA57B78B54704E256024E;
    //mintNFTä¹<âž 2000Uè'ä¹°çš,,ä»£ä,é"Cæ¬ä¹°çš,,äæ°âC
    address public adrBurn = 0xfda48C3f5dDdA69926ef7B3B2AbE73c12e72c95E;
    //ä°æææµ<ç>è'¹|è""æš¼éCæâ†°10%æµ<ç>è'¹çš,,æ¬æ¬¼äæ°âC
    //Transaction fee|Receiving address for 10% fee on staking withdrawal.
    address public adrFee = 0x78Bc9a670323421D040168d1B75543700C1C0450;
    //æ¬æ¬æ± äæ'â¼>ž1%é+£éçš,,20%â^†â^æµ"â¼Cç>™çš,,ä>>ä,ªä,äæé'±âC...äæ°âC ä,šç°;â†•;@âšš
    //20% of the daily pool withdrawal (1%) will be distributed to four different wallet addresses
    upon launch, subject to confirmation.
    address[] public _CEOWallet = [
        0x7a4Df97Ba0Fb3e5bf37Cd0ea720E421a409F0F8f,
        0xb0EAD05769A4f4a76b5644ED97FD6E0A7716323A,
        0xA948c2E97f746e5FbE71912B018A445A00d32dF7,
        0xF9ccEE14e737d83658BA22a1B27A915AAD3E6deA
    ];
}
```

# Smart Contract – Software Analysis

## Function Signatures

```

/**ēĹ·āēē°ā½· */
function oOgetFinanceInfo(
    uint page,
    uint limit,
    uint _type
) public view returns (Finance[] memory, uint) {
    uint total = 0;
    uint skip = page * limit;
    Finance[] memory list = new Finance[] (limit + 1);
    for (uint a = 0; a < limit; a++) {
        //StakeInfo =1  UnStakeInfo=2  MintInfo=3
        if (_type == 1) {
            list[a] = StakeInfo[skip + a];
        } else if (_type == 2) {
            list[a] = UnStakeInfo[skip + a];
        } else if (_type == 3) {
            list[a] = MintInfo[skip + a];
        }
    }
    if (_type == 1) total = StakeInfo.length;
    if (_type == 2) total = UnStakeInfo.length;
    if (_type == 3) total = MintInfo.length;

    return (list, total);
}

```

```

/**ēĹ·āēē°ā½· æ¹æ;² */
function oOgetFinanceInfo2(
    uint start,
    uint end,
    uint _type
) public view returns (Finance[] memory, uint) {
    uint limit = end - start;
    uint total = 0;
    Finance[] memory list = new Finance[] (limit + 1);
    for (uint a = 0; a <= limit; a++) {
        //StakeInfo =1  UnStakeInfo=2  MintInfo=3
        if (_type == 1) {
            list[a] = StakeInfo[start + a];
        } else if (_type == 2) {
            list[a] = UnStakeInfo[start + a];
        }
    }
}

```

```

a2e74af6 => setFeeToSetter (address)
06fdde03 => name ()
95d89b41 => symbol ()
313ce567 => decimals ()
18160ddd => totalSupply ()
70a08231 => balanceOf (address)
dd62ed3e => allowance (address, address)
095ea7b3 => approve (address, uint256)
a9059cbb => transfer (address, uint256)
23b872dd => transferFrom (address, address, uint256)
3644e515 => DOMAIN_SEPARATOR ()
30adf81f => PERMIT_TYPEHASH ()
7ecebe00 => nonces (address)
d505accf => permit (address, address, uint256, uint256, uint8, bytes32, bytes32)
ba9a7a56 => MINIMUM_LIQUIDITY ()
0dfe1681 => token0 ()
d21220a7 => token1 ()
0902flac => getReserves ()
5909c0d5 => price0CumulativeLast ()
5a3d5493 => price1CumulativeLast ()
7464fc3d => kLast ()
6a627842 => mint (address)
89afcb44 => burn (address)
022c0d9f => swap (uint256, uint256, address, bytes)
bc25cf77 => skim (address)
fff6cae9 => sync ()
485cc955 => initialize (address, address)
893d20e8 => getOwner ()
969cd8fe => logTransfer (address, uint256, address, uint256)
fcc55e58 => authorizeCaller (address, bool)
5528ebd7 => authorizeByAuthorized (address)
119df25f => _msgSender ()
8b49d47e => _msgData ()
8da5cb5b => owner ()
715018a6 => renounceOwnership ()
f2fde38b => transferOwnership (address)
b6c52324 => geUnlockTime ()
dd467064 => lock (uint256)
a69df4b5 => unlock ()
d0e30db0 => deposit ()
d6c7cc8b => rewardCurrency ()
0eeca21 => draw ()
b1eac37e => jackpotAmount ()
d847b8e3 => isJackpotReady ()
59992cc8 => setJackpot (uint256)
3d092c6d => checkAndPayJackpot ()
a2340af8 => excludeFromJackpot (address, bool)
992bfe27 => setMaxAttempts (uint256)
9d24dbd5 => setJackpotToCurrency (bool)
688491fc => setJackpotToToken (address, bool)

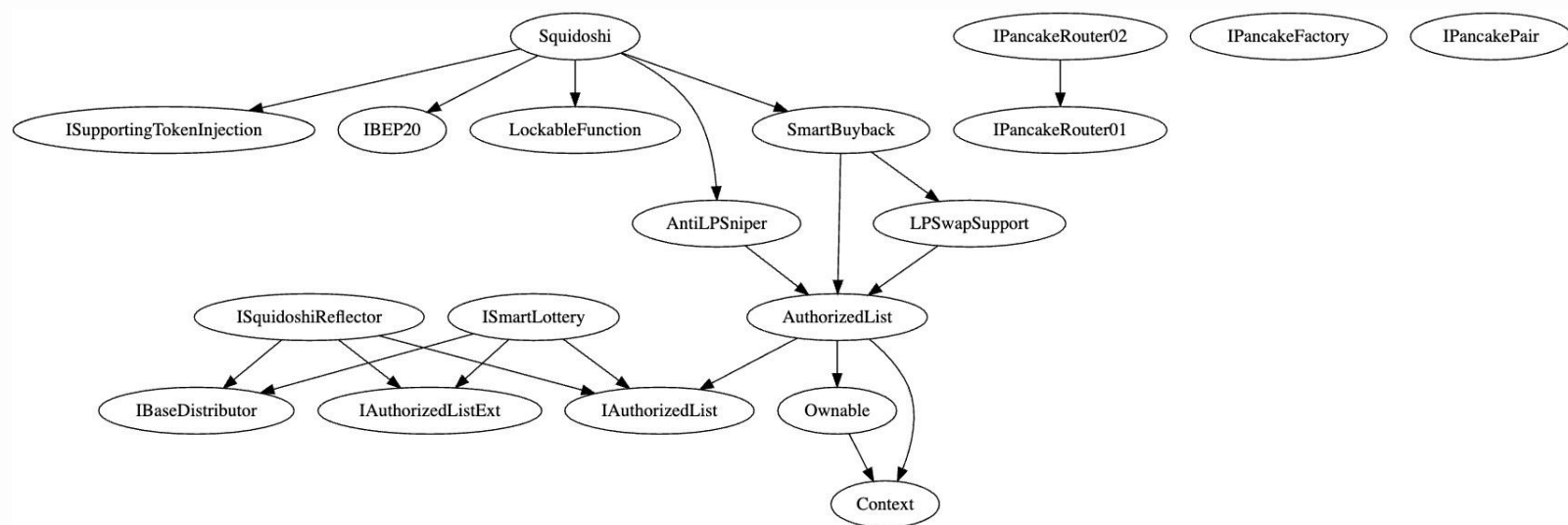
```

```

a55a54e7 => setJackpotEligibilityCriteria(uint256,uint256,uint256)
24a084df => sendValue(address,uint256)
a0b5ffb0 => functionCall(address,bytes)
241b5886 => functionCall(address,bytes,string)
2a011594 => functionCallWithValue(address,bytes,uint256)
d525ab8a => functionCallWithValue(address,bytes,uint256,string)
36455e42 => _functionCallWithValue(address,bytes,uint256,string)
b8544dd7 => renounceAuthorization()
23ac4903 => banHammer(address)
9155e083 => updateBlacklist(address,bool)
6fbff064 => enableAntiSniper(bool)
2d48e896 => setDistributionCriteria(uint256,uint256)
14b6ca96 => setShare(address,uint256)
ffb2c479 => process(uint256)
8f7c9f8c => setRewardToCurrency(bool)
b68a0d25 => setRewardToToken(address,bool)
35c6f9bc => excludeFromReward(address,bool)
edf35253 => claimDividendFor(address)

```

## Inheritance Graph





# Smart Contract – Manual Analysis

Function	Description	Tested	Verdict
<b>Total Supply</b>	provides information about the total token supply	Yes	<b>Passed</b>
<b>Balance Of</b>	provides account balance of the owner's account	Yes	<b>Passed</b>
<b>Transfer</b>	executes transfers of a specified number of tokens to a specified address	Yes	<b>Passed</b>
<b>Approve</b>	allow a spender to withdraw a set number of tokens from a specified account	Yes	<b>Passed</b>
<b>Allowance</b>	returns a set number of tokens from a spender to the owner	Yes	<b>Passed</b>
<b>Buy Back</b>	is an action in which the project buys back its tokens from the existing holders usually at a market price	Yes	<b>Passed</b>
<b>Burn</b>	executes transfers of a specified number of tokens to a burn address	Yes	<b>Passed</b>
<b>Mint</b>	executes creation of a specified number of tokens and adds it to the total supply	NA	NA
<b>Rebase</b>	circulating token supply adjusts (increases or decreases) automatically according to a token's price fluctuations	NA	NA
<b>Blacklist</b>	stops specified wallets from interacting with the smart contract function modules	Yes	<b>Passed</b>
<b>Lock</b>	stops or locks all function modules of the smart contract	Yes	<b>Passed</b>

## Review

- ❖ Active smart contract owner: **0xE1b06313CbeD1447a752b5b96D5a4e36F5E0b508**
- ❖ **Be aware that active smart contract owner privileges constitute an elevated impact to smart contract's safety and security.**
- ❖ **Smart contract can be locked by the owner. This stops or locks all function modules of the smart contract.**
- ❖ **Smart contract owner can blacklist certain wallets from interacting with the contract function modules.**
- ❖ **Smart contract owner can buy back the tokens from the total supply.**
- ❖ **Smart contract uses anti-snipe function module, this may raise the transaction tax, or block an address from doing a transaction.**
- ❖ Owner can-not lock or burn user assets.
- ❖ Owner can-not stop or pause the smart contract.
- ❖ Owner can-not mint tokens after launch.
- ❖ The smart contract utilizes "SafeMath" function to avoid common smart contract vulnerabilities.

```
string private _name = "STAR
TOKEN"; library SafeMath {
function add(uint256 a, uint256 b) internal pure returns (uint256) {
    uint256 c = a + b;
    require(c >= a, "SafeMath: addition overflow");

function sub(uint256 a, uint256 b) internal pure returns (uint256) {
    return sub(a, b, "SafeMath: subtraction overflow");

    uint256 c = a * b;
    require(c / a == b, "SafeMath: multiplication overflow");

    return c;

function div(uint256 a, uint256 b) internal pure returns (uint256) {
    return div(a, b, "SafeMath: division by zero");

function mod(uint256 a, uint256 b) internal pure returns (uint256) {
    return mod(a, b, "SafeMath: modulo by zero");
```

- ❖ The smart contract has low severity issue which may or may not create any functional vulnerability.

```
{  
  "resource": "/STAR TOKEN. sol",  
  "owner": "_generated_diagnostic_collection_name_#0",  
  "severity": 8, (! Low Severity)  
  "Expected token Comma got 'Identifier'",  
  "source": "solc",  
}
```

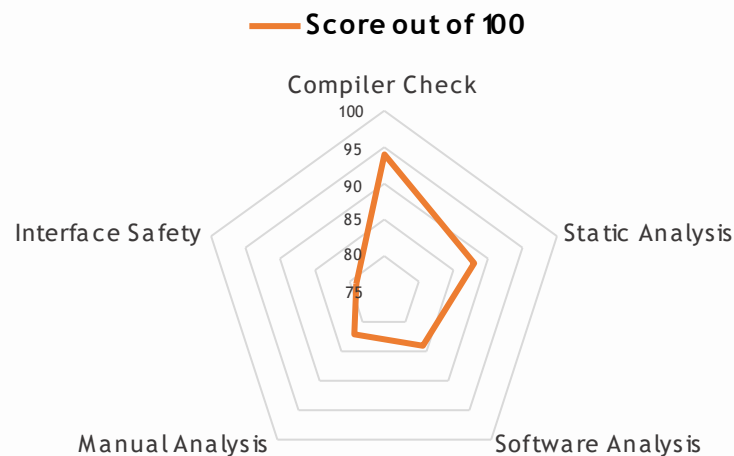
# Smart Contract – SWC Attacks

SWC ID	Description	Verdict
<b>SWC-101</b>	Integer Overflow and Underflow	<b>Passed</b>
<b>SWC-102</b>	Outdated Compiler Version	<b>! Low</b>
<b>SWC-103</b>	Floating Pragma	<b>Passed</b>
<b>SWC-104</b>	Unchecked Call Return Value	<b>Passed</b>
<b>SWC-105</b>	Unprotected Ether Withdrawal	<b>Passed</b>
<b>SWC-106</b>	Unprotected SELFDESTRUCT Instruction	<b>Passed</b>
<b>SWC-107</b>	Re-entrancy	<b>Passed</b>
<b>SWC-108</b>	State Variable Default Visibility	<b>Passed</b>
<b>SWC-109</b>	Uninitialized Storage Pointer	<b>Passed</b>
<b>SWC-110</b>	Assert Violation	<b>Passed</b>
<b>SWC-111</b>	Use of Deprecated Solidity Functions	<b>Passed</b>
<b>SWC-112</b>	Delegate Call to Untrusted Callee	<b>Passed</b>
<b>SWC-113</b>	DoS with Failed Call	<b>Passed</b>
<b>SWC-114</b>	Transaction Order Dependence	<b>Passed</b>
<b>SWC-115</b>	Authorization through tx.origin	<b>Passed</b>
<b>SWC-116</b>	Block values as a proxy for time	<b>Passed</b>
<b>SWC-117</b>	Signature Malleability	<b>Passed</b>
<b>SWC-118</b>	Incorrect Constructor Name	<b>Passed</b>

<b>SWC-119</b>	Shadowing State Variables	<b>Passed</b>
<b>SWC-120</b>	Weak Sources of Randomness from Chain Attributes	<b>Passed</b>
<b>SWC-121</b>	Missing Protection against Signature Replay Attacks	<b>Passed</b>
<b>SWC-122</b>	Lack of Proper Signature Verification	<b>Passed</b>
<b>SWC-123</b>	Requirement Violation	<b>Passed</b>
<b>SWC-124</b>	Write to Arbitrary Storage Location	<b>Passed</b>
<b>SWC-125</b>	Incorrect Inheritance Order	<b>! Low</b>
<b>SWC-126</b>	Insufficient Gas Griefing	<b>Passed</b>
<b>SWC-127</b>	Arbitrary Jump with Function Type Variable	<b>Passed</b>
<b>SWC-128</b>	DoS With Block Gas Limit	<b>Passed</b>
<b>SWC-129</b>	Typographical Error	<b>Passed</b>
<b>SWC-130</b>	Right-To-Left-Override control character (U+202E)	<b>Passed</b>
<b>SWC-131</b>	Presence of unused variables	<b>Passed</b>
<b>SWC-132</b>	Unexpected Ether balance	<b>Passed</b>
<b>SWC-133</b>	Hash Collisions With Multiple Variable Length Arguments	<b>Passed</b>
<b>SWC-134</b>	Message call with hardcoded gas amount	<b>Passed</b>
<b>SWC-135</b>	Code With No Effects (Irrelevant/Dead Code)	<b>Passed</b>
<b>SWC-136</b>	Unencrypted Private Data On-Chain	<b>Passed</b>

# Smart Contract - Risk Status & Radar Chart

Risk Severity	Status
<b>! Critical</b>	None critical severity issues identified
<b>! High</b>	None high severity issues identified
<b>! Medium</b>	None medium severity issues identified
<b>! Low</b>	<b>1 low severity issue identified</b>
<b>Passed</b>	<b>44 functions and instances verified and passed</b>



Compiler Check	94
Static Analysis	88
Software Analysis	84
Manual Analysis	82
Interface Safety	79

## Auditor's Verdict

Defense team has performed a line-by-line manual analysis and automated review of the smart contract. The smart contract was analyzed mainly for common smart contract vulnerabilities, exploits, and manipulation hacks.

**STAR TOKEN's token smart contract source code has LOW RISK**

**SEVERITY. STAR TOKEN has PASSED the smart contract audit.**

### Note for stakeholders

- ❖ Be aware that active smart contract owner privileges constitute an elevated impact on smart contract's safety and security.
- ❖ Make sure that the project team's KYC/identity is verified by an independent firm, e.g., Defense.
- ❖ Always check if the contract's liquidity is locked. A longer liquidity lock plays an important role in project's longevity. It is recommended to have multiple liquidity providers.
- ❖ Examine the unlocked token supply in the owner, developer, or team's private wallets. Understand the project's tokenomics, and make sure the tokens outside of the LP Pair are vested or locked for a longer period of time.
- ❖ Ensure that the project's official website is hosted on a trusted platform, and is using an active SSL certificate. The website's domain should be registered for a longer period of time.

## Important Disclaimer

Defense Network provides contract auditing and project verification services for blockchain projects. The purpose of the audit is to analyse the on-chain smart contract source code, and to provide basic overview of the project. **This report should not be transmitted, disclosed, referred to, or relied upon by any person for any purposes without Defense's prior written consent.**

Defense provides the easy-to-understand assessment of the project, and the smart contract (otherwise known as the source code). The audit makes no statements or warranties on the security of the code. It also cannot be considered as an enough assessment regarding the utility and safety of the code, bug-free status, or any other statements of the contract. While we have used all the data at our disposal to provide the transparent analysis, it is important to note that you should not rely on this report only — we recommend proceeding with several independent audits and a public bug bounty program to ensure the security of smart contracts. **Be aware that smart contracts deployed on a blockchain aren't resistant from external vulnerability, or a hack. Be aware that active smart contract owner privileges constitute an elevated impact to smart contract's safety and security. Therefore, Defense does not guarantee the explicit security of the audited smart contract.**

The analysis of the security is purely based on the smart contracts alone. No applications or operations were reviewed for security. No product code has been reviewed.

**This report should not be considered as an endorsement or disapproval of any project or team.**

The information provided on this report does not constitute investment advice, financial advice, trading advice, or any other sort of advice and you should not treat any of the report's content as such. Do conduct your own due diligence and consult your financial advisor before making any investment decisions.



## About Defense Network

Defense Network provides intelligent blockchain solutions. Defense is developing an ecosystem that is seamless and responsive. Some of our services: Blockchain Security, Token Launchpad, NFT Marketplace, etc. **Defense's mission is to interconnect multiple services like Blockchain Security, DeFi, Gaming, and Marketplace under one ecosystem that is seamless, multi-chain compatible, scalable, secure, fast, responsive, and easy-to-use.**

Defense is built by a decentralized team of UI experts, contributors, engineers, and enthusiasts from all over the world. Our team currently consists of 6+ core team members, and 10+ casual contributors. **Defense provides manual, static, and automatic smart contract analysis, to ensure that project is checked against known attacks and potential vulnerabilities.**

To learn more, visit [www.deepsethi.com/](http://www.deepsethi.com/)

To view our audit portfolio, visit

<https://github.com/techloyaking/Defense-Network->

To book an audit,

