Assure DeFi®

THE VERIFICATION GOLD STANDARD

Security Assessment

ZeussICO

Date: 28/02/2025

Audit Status: FAIL

Audit Edition: Advanced

ASSURE DEFI®

THE VERIFICATION GOLD STANDARD

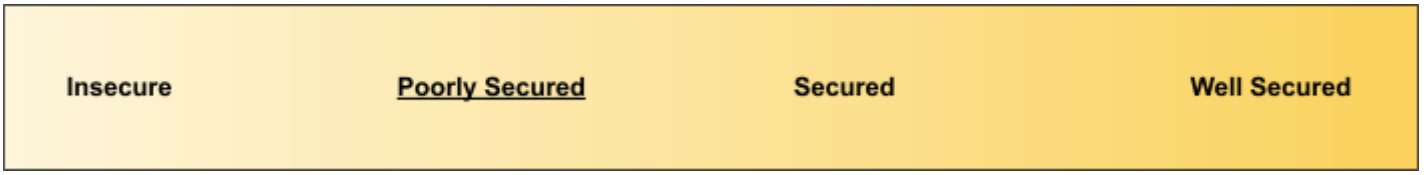
Risk Analysis

Vulnerability summary

Classification	Description
High	High-level vulnerabilities can result in the loss of assets or manipulation of data.
Medium	Medium-level vulnerabilities can be challenging to exploit, but they still have a considerable impact on smart contract execution, such as allowing public access to critical functions.
Low	Low-level vulnerabilities are primarily associated with outdated or unused code snippets that generally do not significantly impact execution, sometimes they can be ignored.
Informational	Informational vulnerabilities, code style violations, and informational statements do not affect smart contract execution and can typically be disregarded.

Executive Summary

According to the Assure assessment, the Customer's smart contract is **Poorly Secured.**



Scope

Target Code And Revision

For this audit, we performed research, investigation, and review of the ZeussICO contracts followed by issue reporting, along with mitigation and remediation instructions outlined in this report.

Target Code And Revision

Project	Assure
Language	Solidity
Codebase	ZeussICO.sol [SHA256] https://etherscan.io/address/0xcb102f03377093f0bdc025346fb29c86441b0164#code
Audit Methodology	Static, Manual

Attacks made to the contract

In order to check for the security of the contract, we tested several attacks in order to make sure that the contract is secure and follows best practices.

Category	Item
Code review & Functional Review	• Compiler warnings. • Race conditions and Reentrancy. Cross-function race conditions. • Possible delays in data delivery. • Oracle calls. • Front running. • Timestamp dependence. • Integer Overflow and Underflow. • DoS with Revert. • DoS with block gas limit. • Methods execution permissions. • Economy model. • Private user data leaks. • Malicious Event log. • Scoping and Declarations. • Uninitialized storage pointers. • Arithmetic accuracy. • Design Logic. • Cross-function race conditions. • Safe Zeppelin module. • Fallback function security. • Overpowered functions / Owner privileges

AUDIT OVERVIEW

HIGH

1. Constructor() – Unsafe Hardcoding of Owner Address

Issue: The contract's constructor sets the owner to a predetermined address rather than dynamically assigning the deployer (i.e., msg.sender) as the owner. This static assignment creates a significant security risk: if the designated owner account is compromised or controlled by a malicious actor, they can exploit the updateToken() function to substitute the legitimate token contract with a malicious ERC20 contract. This, in turn, allows them to withdraw all tokens from the contract using the withdrawUnSoldTokens() function.

Recommendation: Modify the constructor to set the contract owner to the deployer (i.e., msg.sender) rather than a hardcoded address. This change will ensure that the person or entity deploying the contract retains control, thereby reducing the risk of an insider attack and enhancing overall contract security.

MEDIUM

1. Predictable Price Update Enables Front-Running

Issue: The current implementation of the buy() function incorporates a dynamic pricing mechanism that updates the token price at fixed intervals. Because these updates are predictable based on the block timestamp, an attacker can strategically time their purchase to exploit the moment just before the price increases, thereby executing a front-running attack and purchasing tokens at a lower price.

Recommendation: A potential solution is to implement a cooldown mechanism that temporarily delays price updates immediately following a transaction. This would obscure the exact timing of the price change, making it more difficult for attackers to predict and front-run the update. Additionally, incorporating randomness or deferring price adjustments until after transaction confirmation can further reduce this risk.

LOW

1. Input Validation Deficiencies in Constructor and Update Functions

Issue: Several functions within the contract lack adequate input validation for critical parameters. Specifically, the constructor does not verify that the _tokenAddress parameter is non-zero, nor does it ensure that _usdtPrice is greater than zero. Additionally, the updateToken() and updateRecipentAddress() functions do not check that the provided addresses are non-zero. These oversights can lead to unintended behavior if invalid values are supplied.

Recommendation: Add input validation in the constructor to ensure that _tokenAddress is not the zero

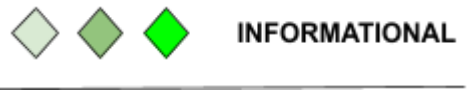
address and that _usdtPrice is greater than zero using require() statements.

Similarly, in the updateToken() and updateRecipentAddress() functions, include checks to guarantee that the supplied addresses are non-zero.

2. Buy() – Recommendation to Implement a Reentrancy Guard

Issue: While the current implementation of the buy() function appears to have a low probability of being exploited through a reentrancy attack, it is still considered best practice to protect against this risk. Adding a reentrancy guard using OpenZeppelin's nonReentrant modifier would provide an additional layer of security by ensuring that the function cannot be called recursively, either directly or indirectly.

Recommendation: Implement OpenZeppelin’s nonReentrant modifier in the buy() function.



1. UpdateSaleStatus() – Missing Event Emission on Sale Closure

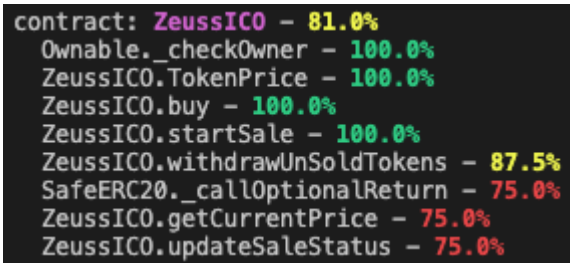
Issue: The updateSaleStatus() function does not emit the SaleStopped() event when the sale is closed. This omission hinders proper tracking of state changes by external monitoring systems and may lead to a lack of transparency for users observing the sale's lifecycle.

Recommendation: Modify the updateSaleStatus() function to emit the SaleStopped() event when the sale is closed.

Testing coverage

During the testing phase, custom use cases were written to cover all the logic of contracts. *Check “Annexes” to see the testing code.*

ZeussICO contract tests:



```
tests/test_zeuss_ico.py::test_start_sale RUNNING
Transaction sent: 0x966958fcc16c4e7d9e5b219f2c50e9131b7e5b93c9950e65321d238f43c3cdf2
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 0
  ERC20Mock.constructor confirmed   Block: 1   Gas used: 702419 (5.85%)
  ERC20Mock deployed at: 0x3194cBDC3dbcd3E11a07892e7bA5c3394048Cc87

Transaction sent: 0x83a350c1cd3bb5136577c61da200b31fc3a8a527686e181e0f3cbf7856f88880
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 1
  ZeussICO.constructor confirmed   Block: 2   Gas used: 1154821 (9.62%)
  ZeussICO deployed at: 0x602C71e4DAC47a042Ee7f46E0aee17F94A3bA0B6

Transaction sent: 0x2660bc44489d077e3a540632a1ef1f7352ce23cef486cf293a7610827da6dbd9
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 0
  ZeussICO.startSale confirmed (reverted)   Block: 3   Gas used: 22178 (0.18%)

Transaction sent: 0xb6afb8581defe5c1906c435d8190d601840d2ff81cc02285d4b1e16c3b11b14
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 0
  ZeussICO.startSale confirmed   Block: 4   Gas used: 50375 (0.42%)

Transaction sent: 0x28a96b1d6ab4a8a6dbc35a2a2707e10d7da2794402e93e2c742532f22b05762b
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 1
  ZeussICO.startSale confirmed (Sale is already started)   Block: 5   Gas used: 23067 (0.19%)

tests/test_zeuss_ico.py::test_start_sale PASSED
tests/test_zeuss_ico.py::test_update_sale_status RUNNING
Transaction sent: 0xfb56fde70e3a517420f3be0fbb7605aacd681ae32bea385c73eebaed6d1f74e5
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 2
  ERC20Mock.constructor confirmed   Block: 6   Gas used: 702419 (5.85%)
  ERC20Mock deployed at: 0xE7eD6747FaC5360f88a2EFC03E00d25789F69291

Transaction sent: 0x0b850b160325b5980278079e22fa7259f1dd1e6f1ab0462cfdf9fe32ac53bfe2
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 3
  ZeussICO.constructor confirmed   Block: 7   Gas used: 1154809 (9.62%)
  ZeussICO deployed at: 0x6951b5Bd815043E3F842c1b026b0Fa888Cc2DD85

Transaction sent: 0xdd6160a6f594738d2c014cd1af7450159a01bae1b7bc09bcd6cf8c57565ab824
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 1
  ZeussICO.updateSaleStatus confirmed (reverted)   Block: 8   Gas used: 22408 (0.19%)

Transaction sent: 0xf720f8e61cdb1592538fd2eae934eafc0dbf2d52b5d824749c3a0ad131cde8ca
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 2
  ZeussICO.updateSaleStatus confirmed   Block: 9   Gas used: 49113 (0.41%)

Transaction sent: 0x68ed079cc6fa246e66759401401649aaf63dda9935f11b475b9db2344e920944
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 3
  ZeussICO.startSale confirmed (Sale is already started)   Block: 10   Gas used: 23067 (0.19%)

tests/test_zeuss_ico.py::test_update_sale_status PASSED
tests/test_zeuss_ico.py::test_update_token RUNNING
Transaction sent: 0x5dc9de0ff1a7d45ebe72d95344d737c6fb02a71451a884f7e98aa0f090a512cf
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 4
  ERC20Mock.constructor confirmed   Block: 11   Gas used: 702419 (5.85%)
  ERC20Mock deployed at: 0xe0aA552A10d7EC8760Fc6c246D391E698a82dDf9

Transaction sent: 0x713f09726fbb3f702e724c5a4e671327c46b6367f393d5562608eea94a898f23
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 5
  ERC20Mock.constructor confirmed   Block: 12   Gas used: 702419 (5.85%)
  ERC20Mock deployed at: 0x6b4BDe1086912A6Cb24ce3dB43b3466e6c72AFd3

Transaction sent: 0x7e738386feaff9401953c69279edb97e6d658338f1be9ebc678dfccc5fe6d804
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 6
  ZeussICO.constructor confirmed   Block: 13   Gas used: 1154821 (9.62%)
  ZeussICO deployed at: 0x9E4c14403d7d9A8A782044E86a93CAE09D7B2ac9

Transaction sent: 0x7127aba7e61d37177e9d2c98f20f4b282883bbb4bb667b764a15f1a538fea67f
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 2
  ZeussICO.updateToken confirmed (reverted)   Block: 14   Gas used: 22720 (0.19%)

Transaction sent: 0xd292d7dc68bb5129a65570aa6720ea906afe408e9021c02c4f9e22d29fd2044b
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 4
  ZeussICO.updateToken confirmed   Block: 15   Gas used: 28528 (0.24%)

tests/test_zeuss_ico.py::test_update_token PASSED
```

```
tests/test_zeuss_ico.py::test_update_recp_addr RUNNING
Transaction sent: 0x94fd3b19ad9c42727d5909a722e157b2d747bc12c3255cb11a490250b479c2f4
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 7
  ERC20Mock.constructor confirmed   Block: 16   Gas used: 702419 (5.85%)
  ERC20Mock deployed at: 0xcCB53c9429d32594F404d01fbe9E65ED1DCda8D9

Transaction sent: 0xa560b4e67874e9e5e8322b3f8f3d2659f70d410761d744b355c72749582ca2ca
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 8
  ZeussICO.constructor confirmed   Block: 17   Gas used: 1154821 (9.62%)
  ZeussICO deployed at: 0x420b1099B9eF5baba6D92029594eF45E19A04A4A

Transaction sent: 0xe90f34f81d16e28affa9a43e2e8743516e177f163ad2ff8e59eb71be07849905
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 3
  ZeussICO.updateRecipentAddress confirmed (reverted)   Block: 18   Gas used: 22751 (0.19%)

Transaction sent: 0xd944edeea57bd575ef74af3b895b4998fbfd84bcce1f7aef00f225c3e6f230be
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 5
  ZeussICO.updateRecipentAddress confirmed   Block: 19   Gas used: 28559 (0.24%)

tests/test_zeuss_ico.py::test_update_recp_addr PASSED
```

```
tests/test_zeuss_ico.py::test_buy RUNNING
Transaction sent: 0x967aba4584a2053b3b3926c078c74b0089af8b176295771db1f2a506f6a49494
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 9
  ERC20Mock.constructor confirmed   Block: 20   Gas used: 702419 (5.85%)
  ERC20Mock deployed at: 0xa3B53dDCd2E3fC28e8E130288F2aBD8d5EE37472

Transaction sent: 0x15fdd0a88ee230fd23bcbaa412115c6bce4ff579a54201fc8fd4006111d27e64
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 10
  ERC20Mock.constructor confirmed   Block: 21   Gas used: 702419 (5.85%)
  ERC20Mock deployed at: 0xb6286fAFd0451320ad6A8143089b216C2152c025

Transaction sent: 0x73e9b439cf3df596222dfded7dadc6fcdab6b165dfff5c1a12d2863170118b6b
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 11
  ZeussICO.constructor confirmed   Block: 22   Gas used: 1154821 (9.62%)
  ZeussICO deployed at: 0x7a3d735ee6873f17Dbdcab1d51B604928dc10d92

Transaction sent: 0xf93a9499d3a179f8f0f723ca38331aa717a34df6acef784531d5a2522100461e
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 4
  ZeussICO.buy confirmed (Sale is Closed)   Block: 23   Gas used: 22714 (0.19%)

Transaction sent: 0x1e8a5ad54de482d2f2e73f5286d3513c2173225a02a647b7b5f3920af225efdf
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 6
  ZeussICO.startSale confirmed   Block: 24   Gas used: 50375 (0.42%)

Transaction sent: 0xc80d23f04503248b27b37faff90772dfd5519099e5cc26dda5fcfbb78ec7dd60
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 7
  ZeussICO.setTokenAddr confirmed   Block: 25   Gas used: 28569 (0.24%)

Transaction sent: 0x0ee9742bbd9dc8ceb3b19424c1de27e725c25206f92ce869b6e47e4a0c88f26e
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 12
  ERC20Mock.mint confirmed   Block: 26   Gas used: 65660 (0.55%)

Transaction sent: 0xad57009be88e93b7f1f19145e09c301845057af09cd3e51df830080540302155
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 13
  ERC20Mock.mint confirmed   Block: 27   Gas used: 50648 (0.42%)

Transaction sent: 0x7b5007cba2522de0e530d18ccbc24b6f8a302cf196378b59c91c3e1153466404
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 5
  ERC20Mock.approve confirmed   Block: 28   Gas used: 44161 (0.37%)

Transaction sent: 0xf6784d1a2c221fc7efde8725ecc36941aef928197ebdc1c7ce328ed8ef4d9475
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 0
  ERC20Mock.approve confirmed   Block: 29   Gas used: 44161 (0.37%)

Transaction sent: 0x5fb396a82bc21b8726db17495e8cdacc2aa3877721ed9dedbfe7eae20ef3a5b2
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 6
  ZeussICO.buy confirmed (Insufficient token balance)   Block: 30   Gas used: 95071 (0.79%)

Transaction sent: 0x0cd6baa49fecc1c174760e2ab8e445500eab3c335a88ee85727fc29cacc08b6b
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 14
  ERC20Mock.mint confirmed   Block: 31   Gas used: 65660 (0.55%)

Transaction sent: 0x7c197e949c3a49729934584c32a08bf3f297731dff67e33e8d988a73ad518890
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 7
  ZeussICO.buy confirmed   Block: 32   Gas used: 284925 (2.37%)

Transaction sent: 0x487916a731a78015bbbfa19c9a048856e0036a36eeafb97c5fff3fd0279d1eda
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 1
  ZeussICO.buy confirmed   Block: 33   Gas used: 224925 (1.87%)

Transaction sent: 0xd8cc0be7698764793d33dcbaef411ab009b63e09c7f1adf0d00dab5c55d519b8
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 8
  ZeussICO.buy confirmed   Block: 35   Gas used: 207484 (1.73%)

tests/test_zeuss_ico.py::test_buy PASSED
```

```
tests/test_zeuss_ico.py::test_withdraw_unsold RUNNING
Transaction sent: 0xffaafde263215cb169ea2edd4cd2adc8721f2c03a5677982daf2426651c06dd5
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 15
  ERC20Mock.constructor confirmed   Block: 36   Gas used: 702419 (5.85%)
  ERC20Mock deployed at: 0x30375B532345B01cB8c2AD12541b09E9Aa53A93d

Transaction sent: 0x0a75f15520ea11ee87bfb7c9d9b41e3a73e7dec5436a9e6fba7be35cbb8b0a6d
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 16
  ZeussICO.constructor confirmed   Block: 37   Gas used: 1154821 (9.62%)
  ZeussICO deployed at: 0x26f15335BB1C6a4C0B660eDd694a0555A9F1cce3

Transaction sent: 0x111fa3a7958502363cb27bddc24064fad9be98dda9e01c87a69080d7c2c81cc4
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 9
  ZeussICO.withdrawUnSoldTokens confirmed (reverted)   Block: 38   Gas used: 22885 (0.19%)

Transaction sent: 0x7e7850a5b0363394d28698a8c171c56b8c2993ce57952a0a308f137e7a697d9d
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 8
  ZeussICO.withdrawUnSoldTokens confirmed (Not enough balance)   Block: 39   Gas used: 22897 (0.19%)

Transaction sent: 0xcc31dce3e6598e735d39469d70f9cd7ef523077de003945c6baa0130f3fabbf9
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 17
  ERC20Mock.mint confirmed   Block: 40   Gas used: 65660 (0.55%)

Transaction sent: 0x259952351f6a70d86916d9fbbde00340d254a284c198c608af50de639a922112
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 9
  ZeussICO.withdrawUnSoldTokens confirmed   Block: 41   Gas used: 57199 (0.48%)

tests/test_zeuss_ico.py::test_withdraw_unsold PASSED
```

Annexes

Testing code:

ZeussICO:

```
from brownie import (
    reverts,
)


from scripts.helpful_scripts import (
    ZERO_ADDRESS,
    DAY_TIMESTAMP,
    get_account,
    increase_timestamp
)


from scripts.deploy import (
    deploy_mock_erc,
    deploy_zeuss_ico
)


def test_start_sale(only_local):
    # Arrange
    owner = get_account(0)
    other = get_account(1)


    contract_owner = "0x2cc312F73F34BcdADa7d7589CB3074c7Dc06ebE9"
    token = deploy_mock_erc(owner)
```

```
    ico = deploy_zeuss_ico(owner, token.address, 1000000)
    with reverts():
        ico.startSale({"from": other})
    tx = ico.startSale({"from": contract_owner})
    assert tx.events['SaleStarted'] is not None
    with reverts("Sale is already started"):
        ico.startSale({"from": contract_owner})


def test_update_sale_status(only_local):
    # Arrange
    owner = get_account(0)
    other = get_account(1)


    contract_owner = "0x2cc312F73F34BcdADa7d7589CB3074c7Dc06ebE9"
    token = deploy_mock_erc(owner)


    ico = deploy_zeuss_ico(owner, token.address, 1000000)
    with reverts():
        ico.updateSaleStatus(True, {"from": other})
    ico.updateSaleStatus(True, {"from": contract_owner})
    with reverts("Sale is already started"):
        ico.startSale({"from": contract_owner})



def test_update_token(only_local):
    # Arrange
    owner = get_account(0)
    other = get_account(1)
```

```
    contract_owner = "0x2cc312F73F34BcdADa7d7589CB3074c7Dc06ebE9"
    token = deploy_mock_erc(owner)
    new_token = deploy_mock_erc(owner)

    ico = deploy_zeuss_ico(owner, token.address, 1000000)
    with reverts():
        ico.updateToken(new_token.address, {"from": other})

    assert ico.tokenAddress() == token.address
    ico.updateToken(new_token.address, {"from": contract_owner})
    assert ico.tokenAddress() == new_token.address

def test_update_recp_addr(only_local):
    # Arrange
    owner = get_account(0)
    other = get_account(1)
    extra = get_account(2)

    contract_owner = "0x2cc312F73F34BcdADa7d7589CB3074c7Dc06ebE9"
    token = deploy_mock_erc(owner)

    ico = deploy_zeuss_ico(owner, token.address, 1000000)
    with reverts():
        ico.updateRecipentAddress(extra, {"from": other})

    assert ico.fundsRecipentAddress() == "0xc728595c1Ae60DfA2Db7F20BBFDbEf649d7c2783"
    ico.updateRecipentAddress(extra, {"from": contract_owner})
```

```
    assert ico.fundsRecipentAddress() == extra


def test_buy(only_local):
    # Arrange
    owner = get_account(0)
    other = get_account(1)
    extra = get_account(2)


    contract_owner = "0x2cc312F73F34BcdADa7d7589CB3074c7Dc06ebE9"
    token = deploy_mock_erc(owner)
    mock_usdt = deploy_mock_erc(owner)
    token_price = 1000000


    ico = deploy_zeuss_ico(owner, token.address, token_price)
    with reverts("Sale is Closed"):
        ico.buy(1e18, 0, {"from": other})
    ico.startSale({"from": contract_owner})
    # created mock function just for testing purpose
    ico.setTokenAddr(mock_usdt.address, {"from": contract_owner})
    # mint some tokens
    mock_usdt.mint(other, 5e18)
    mock_usdt.mint(extra, 5e18)
    mock_usdt.approve(ico.address, 5e18, {"from": other})
    mock_usdt.approve(ico.address, 5e18, {"from": extra})


    with reverts("Insufficient token balance"):
        ico.buy(1e18, 0, {"from": other})
    token.mint(ico.address, 10e18)
```

```
    tx = ico.buy(1000000, 0, {"from": other})
    assert tx.events['TokenPurchased'][0]['user'] == other
    assert tx.events['TokenPurchased'][0]['amountPaid'] == 1000000
    assert tx.events['TokenPurchased'][0]['tokensReceived'] == 0.98e18
    assert tx.events['TokenPurchased'][0]['pricePerToken'] == token_price

    tx = ico.buy(3000000, 0, {"from": extra})
    assert tx.events['TokenPurchased'][0]['user'] == extra
    assert tx.events['TokenPurchased'][0]['amountPaid'] == 3000000
    assert tx.events['TokenPurchased'][0]['tokensReceived'] == 2.94e18
    assert tx.events['TokenPurchased'][0]['pricePerToken'] == token_price

    increase_timestamp(15 * DAY_TIMESTAMP)

    tx = ico.buy(1000000, 0, {"from": other})
    assert tx.events['TokenPurchased'][0]['user'] == other
    assert tx.events['TokenPurchased'][0]['amountPaid'] == 1000000
    assert tx.events['TokenPurchased'][0]['pricePerToken'] == 1200000

def test_withdraw_unsold(only_local):
    # Arrange
    owner = get_account(0)
    other = get_account(1)
    extra = get_account(2)

    contract_owner = "0x2cc312F73F34BcdADa7d7589CB3074c7Dc06ebE9"
    token = deploy_mock_erc(owner)
```

```
ico = deploy_zeuss_ico(owner, token.address, 1000000)

with reverts():

    ico.withdrawUnSoldTokens(extra, 1e18, {"from": other})

with reverts("Not enough balance"):

    ico.withdrawUnSoldTokens(extra, 0, {"from": contract_owner})

# mint some tokens

token.mint(ico.address, 5e18)

tx = ico.withdrawUnSoldTokens(extra, 1e18, {"from": contract_owner})

assert tx.events['Transfer'][0]['from'] == ico.address

assert tx.events['Transfer'][0]['to'] == extra

assert tx.events['Transfer'][0]['value'] == 1e18
```

Technical Findings Summary

Findings

Vulnerability Level	Total	Pending	Not Apply	Acknowledged	Partially Fixed	Fixed
High	1					
Medium	1					
Low	2					
Informational	1					

Assessment Results

Score Results

Review	Score
Global Score	**70/100**
Assure KYC	Not completed
Audit Score	70/100

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 84 Points for a higher standard, if a project does not attain 85% is an automatic failure. Read our notes and final assessment below. The Global Score is a combination of the evaluations obtained between having or not having KYC and the type of contract audited together with its manual audit.

Audit Failed

Following our comprehensive security audit of the token contract for the ZeussICO project, the project did not fulfill the necessary criteria required to pass the security audit.

Disclaimer

Assure Defi 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 advocating for the Project, and users relying on this report should not consider this as having any merit for financial adZeussICO 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 adZeussICO, nor should it be used to signal that any person reading this report should invest their funds without sufficient individual due diligence, regardless of the findings presented. Information is provided ‘as is, and Assure Defi is under no covenant to audit completeness, accuracy, or solidity of the contracts. In no event will Assure Defi 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 serZeussICOs provided by Assure Defi are subject to dependencies and are under continuing development. You agree that your access or use, including but not limited to any serZeussICOs, 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 serZeussICOs may access, and depend upon, multiple layers of third parties.