# Margin Token Audit Information

## 1. Introduction

Margin Tokens allow for holders to gain exposure to a dual momentum trading system acting on a portfolio of tokens, simply by holding the Margin Token in their Metamask wallet. Each Margin Token is 100% backed by its underlying assets. To purchase a Margin Token, supply one of the supported assets to the Margin Token Portfolio. Send a Margin Token back to the Margin Token Portfolio to redeem a supported token of your choice.

If a token, such as BNB, is in a strong uptrend and therefore desired in the Margin Token Portfolio, there will be a reward for supplying this asset and a fee for redeeming it. If the market turns and BNB is no longer desired, there will be a reward for redeeming this asset and a fee for supplying it. This creates a potential arbitrage opportunity that incentivizes market participants to keep the pool's allocations consistent with the dual momentum trading strategy.

An automated dual momentum trading system will update the deposit and redeem fees or rewards based on its strategy. Rather than the pool of assets incurring trading fees internally by interacting with a DeFi protocol such as PancakeSwap, the fees are passed on to the users via the rewards for depositing and redeeming assets as desired.

Margin Tokens build off Trend Tokens by having the ability to supply assets to Venus, and use that as collateral to borrow other assets such as USDT which can be swapped for assets such as BTC to gain added exposure to it. The dual momentum system updates the Margin Token portfolio on its desired holdings in the contract and in Venus including its collateral and borrow positions.

There will be types of Margin Tokens, each with a different portfolio strategies. This is similar to the different types of portfolio options offered for Trendbot and Marginbot. To begin, three Trend Tokens will be offered.

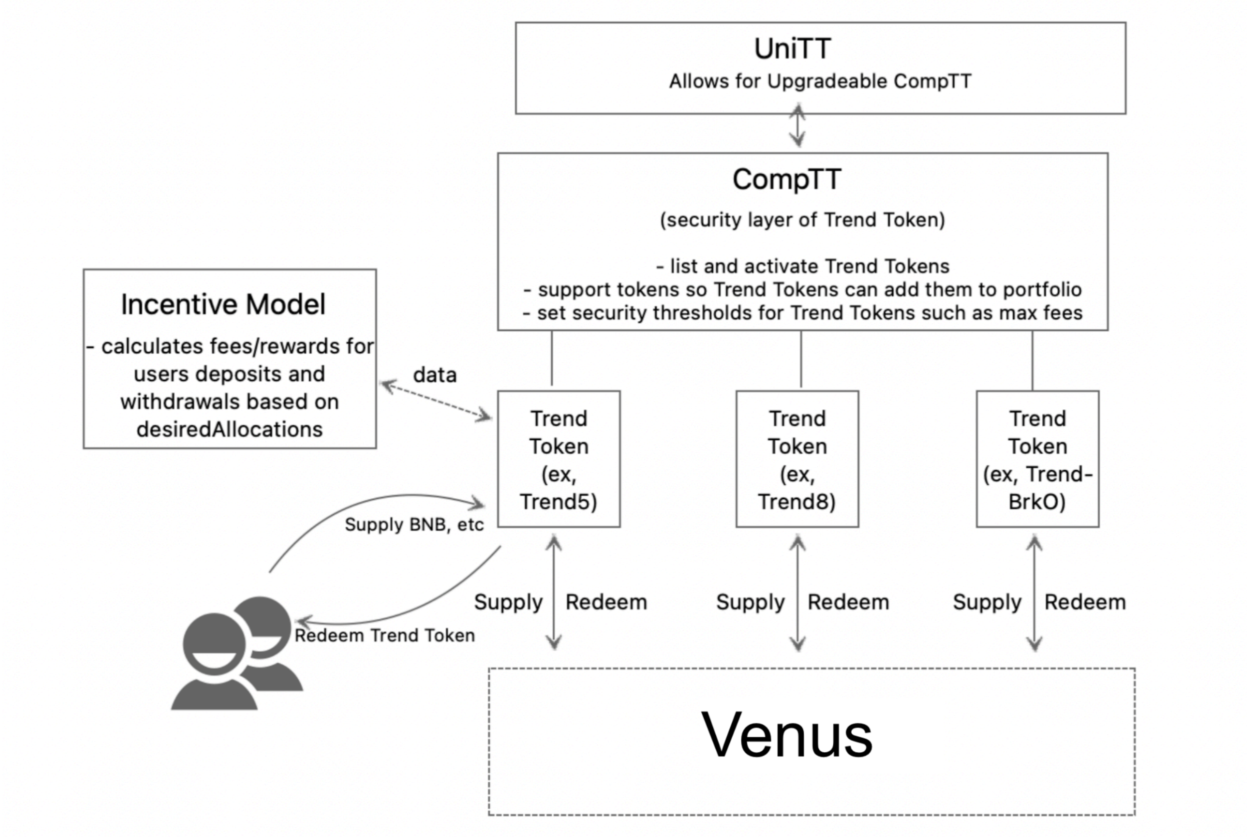
1) **MARGIN5:** Portfolio of USDT, BNB, and 3 top tokens by liquidity on Binance.
2) **MARGIN8:** Portfolio of USDT, BNB, and 5 top tokens by liquidity on Binance.
3) **MARGIN-BrkO:** Portfolio of USDT, BNB, and 0-5 tokens making new all time highs out of the highest 10-20 liquid tokens on Binance.

# 2. Architecture

## 2.1 Terminology and Overview

CompTT: governing contract over all Margin Token portfolios
Margin Token Portfolio: the smart contract holding all assets and issues Margin Tokens
Margin Token: a token representing ownership of the Margin Token Portfolio
IncentiveModel: calculates fees and amounts out when buying, selling, or swapping with the Margin Token Portfolio
Venus: lending and borrowing DeFi protocol that the Margin Token Portfolio connects to

## 2.2 Overall Architecture



UniTT
Allows for Upgradeable CompTT
CompTT
(security layer of Trend Token)
- list and activate Trend Tokens
- support tokens so Trend Tokens can add them to portfolio
- set security thresholds for Trend Tokens such as max fees
Incentive Model
- calculates fees/rewards for users deposits and withdrawals based on desiredAllocations
data
Trend Token (ex, Trend5)
Trend Token (ex, Trend8)
Trend Token (ex, Trend-BrkO)
Supply BNB, etc
Redeem Trend Token
Supply
Redeem
Supply
Redeem
Supply
Redeem
Venus

# 3.0 Key Files

The main files of the Margin Token Ecosystem include:

1. CompTT.sol
2. MarginToken.sol
3. IncentiveModel.sol

## 3.1 CompTT.sol

The Comptroller (CompTT) governs all Margin Tokens and limits their behavior as an added layer of security. The management of Margin Tokens may be different from the management of the Comptroller so this reduces the risk of a Margin Token manager intentionally or unintentionally harming their Margin Token holders. For example, accidentally setting desired borrow positions that would liquidate its holdings on Venus and suffering the liquidation fee.

### 3.1.1 Restrictions CompTT has for specific Margin Tokens (i.e MARGIN10)

| Restriction | Description |
|---|---|
| isLocked | Allows manager to make key changes such as change CompTT, IncentiveModelSimple, manager, tradingBot, fees, max supply |
| isActive | Allows no Margin Token (deposits, redeems, trades) or Venus (supply, redeem, borrow, repay) actions. May be used in an emergency |
| isDeposit | Allows for users to buy Trend Tokens. May be used if CompTT management does not want Margin Token to gain AUM |
| isRedeem | Allows for users to sell Trend Tokens. May be used if CompTT management |
| isTrade | Allows for users to swap one token (BTCB) for another (ETH) using Trend Token portfolio |
| maxTradeFee | The maximum trade fee this Trend Token can have |
| maxPerformanceFee | The maximum performance fee this Trend Token can have |
| maxDisableValue | Maximum value (contract + collateral + abs(borrow)) to disable token from portfolio. This prevents disabling a token with large values and therefore experiencing a sharp decrease in Margin Token price since this price is calculated by ‘equity in portfolio tokens / trend token supply’ |
| isSupplyVenus | Allows for Trend Token to supply assets to Venus |
| isBorrowVenus | Allows for Trend Token to borrow from Venus |
| maxBorrowFactor | Sets the maximum amount the Margin Token may borrow relative to its total borrowable amount. Example, if 100k is supplied @80 collateral factor, the Margin Token may borrow 80k. A maxBorrowFactor of 50% would allow the Margin Token to borrow up to 40k. |
| maxMargin | Sets the maximum amount of margin the Margin Token can have. Similar to maxBorrowFactor except no reliance on borrowable amounts. Example, if there is 100k of equity (assets in contract + collateral - borrow) and maxMargin at 50%, the Margin Token may set positions to borrow up to 50k. |

### 3.1.2 Restriction CompTT has for specific tokens (i.e BTCB, BNB, USDT, etc) across all Margin Tokens

| Restriction | Description |
|---|---|
| isActive | Prevents any activity with this token including user actions (deposit, redeem, trade) or venus actions (deposit, redeem, borrow, repay). Used in case of an emergency such as price oracle affected |
| isDeposit | Allows token to be deposited in Margin Token portfolio for a Margin Token (MARGIN10) |
| isRedeem | Allows token to be redeemed from Margin Token portfolio for a Margin Token |
| isTrade | Allows token to be traded (bought or sold) using asset in Margin Token portfolio |
| vToken | Non-zero if vToken has been supported (allows for potential venus interactions) |
| isVenusActive | True if vToken is currently active (allows for venus interactions) |
| isSupplyActive | Allows token to be supplied to Venus for passive income and used as collateral |
| isBorrowActive | Allows token to be borrowed from Venus by Margin Token portfolio |

Margin Tokens portfolio must get permission from CompTT to perform a variety of actions including adding new tokens to the portfolio, users buying and selling Margin Tokens, users selling one asset for another, and any interaction with Venus. The above restrictions are checked to determine if a Margin Token has permission to perform the desired action.

### pauseGaurdian

The pauseGaurdian is able to pause the entire ecosystem (Comptroller + all Trend Tokens or Margin Tokens) as well as pause an individual Margin Token, or pause the use of an asset (SOL) across all Trend Tokens. Only the admin can unpause. It will be controlled by a monitoring system and pause operations if anything is wrong (price oracle manipulation)

### Admin

The admin address is the most powerful in the ecosystem. It has the ability to control all factors in 3.1.1 and 3.1.2. It may be controlled by XTT token holders in the future.

### LockedWallet

In order for the admin to make any changes, the state must be unlocked. This provides a layer of security in the event the admin keys are compromised.

### 3.1.3 Overview of Management

‘admin’ may

- Support new Venus comptroller
- Set BNB (wbnb and vbnb addresses)
- Set new price oracle
- Support new Trend Tokens and underlying assets
- Change pauseGaurdian
- 3.3.1 and 3.3.2 above

‘pauseGaurdian’ may:

- Pause state of entire Margin Token ecosystem

‘lockedWallet’ may:

- Lock and unlock state of comptroller to prevent any changes made by admin (security layer if admin keys are compromised)
- Change lockedWallet address

## 3.2 MarginToken.sol

MarginToken.sol provides three main functions;

### 3.2.1 Update desired position sizes

- tradingBot address is controlled by an off-chain trend following trading bot that calculates desired position sizes
- Every 6 hours the off-chain trading bot updates the desired position sizes
- A variety of checks are in place to ensure the desired position sizes are reasonable and are safe, including:
    a) Net positions (contract+collateral+borrow) equals 100%
    b) Desired borrows does not cause risk of liquidation and there is a sufficient supply of collateral

### 3.2.2 Allow users to buy, sell, and trade

- Users may buy Margin Tokens, for example with BNB
- Users may sell Margin Tokens, for example for BNB
- Users may swap one token (BNB) for another (USDT) using assets in the Margin Token portfolio
- IncentiveModelSimple.sol calculates rewards and fees to incentivize users to deposit and remove assets according to the desired position sizes in step 1 above.

    - For example, if there is 0% USDT in the Margin Token portfolio, users will earn a reward of approximately 0.10% for selling USDT to the portfolio and a fee of 0.10% of buying USDT from the portfolio

#### 3.2.3 Allow *tradingBot* to supply, redeem, borrow, and repay assets with Venus

- The public may perform this action if *openVenus* = true
- Margin Token's portfolio must get permission from *CompTT* to execute any actions. The *CompTT* will ensure:
    a) the action brings the the current *Venus* positions closer to the desired
    b) any redeem or borrow does not bring the Margin Token too close to liquidation as determined by the *maxBorrowFactor*

#### tradingBot

The tradingBot keys have limited ability to update desired position sizes and change the portfolio, and adjust the Venus collateral and borrow as permitted by the CompTT. These keys will be controlled by an off-chain algorithm that dictates the desired position sizes according to a momentum based strategy.

#### Manager

The manager keys have flexibility to make high security changes for a given Margin Token. Although the CompTT must have its state unlocked in order for the manager to make any changes, many changes have restrictions (performance fees).

### 3.3 IncentiveModelSimple.sol

- Fees, rewards, discounts

## 4.0 Deploy and Configuration Instructions

March 16th:
CompTT: 0x219928ddfF4A2655f237660A36725A2DFCe2F8a7
MarginToken: 0x76591d3ad73c1bFAeE505F42f1930695aF3626A7

#### 4.1 Comptroller Deployment and Configuration

Governs all deployed Trend Tokens and ensures safety parameters

1. Deploy XTT from XTTgov.sol
   Result Testnet: 0x3fF5f7ca6257E29deD56180f12Dd668c4D4b8ad3
   Result Mainnet: 0x6e568e89Dc8c68ca3f6d918f30c75A081a14C06E

2. Paste XTT address in CompTT.sol (line 126)
   Input Testnet: 0x3fF5f7ca6257E29deD56180f12Dd668c4D4b8ad3
   Input Mainnet: 0x6e568e89Dc8c68ca3f6d918f30c75A081a14C06E

3. Deploy CompTT and UniTT:
   Result compTT: 0xcb2BfFA12d67A71D68366fa0F247dd4BAdccD53a
   Result UniTT: 0x9195e05DA23a1DD6D48eFa71d2259E4E8C8Be642

   Result UniTT mainnet: 0xDed21cdB9831B8002cC2BfaBc4D058a1CE54e074
   Result CompTT mainnet: 0x45Da13dC0A300908E7bb457cf9730954b1c0ea65

4. _updateLockedState() in CompTT:
   Input: false

5. Configure CompTT and UnitTT
   - _setPendingImplementation() in UniTT using CompTT address
   - _updateLockedState() in CompTT to ‘false’
   - _become() in CompTT using UniTT address
   - use UniTT address but CompTT ABI

6. Deploy ChainlinkOracle.sol
   Input Testnet: 0x9195e05DA23a1DD6D48eFa71d2259E4E8C8Be642,86400
   Result Testnet: 0xcC05B31B393504DbE4767109c80DE0a6c501b7a6 (different admin)

   Input Mainnet:
   Result Mainnet: 0x057b84fa2cdC0EE47b4d9b4bDdc2d9F68654fF3e

7. _setPriceOracle in CompTT
   Input testnet: 0xcC05B31B393504DbE4767109c80DE0a6c501b7a6
   Input Mainnet: 0x057b84fa2cdC0EE47b4d9b4bDdc2d9F68654fF3e (deployed Mar 20)

8. _setVenusComp() using Venus Comptroller
   Input Testnet: 0x94d1820b2D1c7c7452A163983Dc888CEC546b77D
   Input Mainnet: 0xfD36E2c2a6789Db23113685031d7F16329158384

9. _setBNB() using wbnb/vbnb addresses
   Input Test: 0xae13d989daC2f0dEbFf460aC112a837C89BAa7cd,0x2E7222e51c0f6e98610A1543Aa3836E092CDe62c
   Input Main: 0xbb4CdB9CBd36B01bD1cBaEBF2De08d9173bc095c,0xA07c5b74C9B40447a954e1466938b865b6BBea36

### 4.2 Margin Tokens Deployment and Configuration

1. Deploy MarginToken from MarginToken.sol
   Input testnet: 0x9195e05DA23a1DD6D48eFa71d2259E4E8C8Be642 (from CompTT 4.2 above)
   Result testnet: 0xc86bC58164Fa11Bb00AA593B2C7D70B8fcaeB5F8

Input main: 0xDed21cdB9831B8002cC2BfaBc4D058a1CE54e074
Result main: 0x09F94FC974F2E9dFD3c286Ca290Cd350E319a967

2. _supportTrendToken() in CompTT
   Input testnet: 0xcdEF3f97b29474dc206b8E35a8791Bd35bf0FDc3
   Input mainnet: 0x09F94FC974F2E9dFD3c286Ca290Cd350E319a967

3. Deploy IncentiveModel.sol
   Result testnet: 0x18B2f810B3592E4Cf4bC2DC0FB1317536e75dCC5
   Result mainnet: 0x4C1598892E3EEEb9dE18CdaE7698545EC0680f7F

4. _updateIncentivesModel( addr) in MarginToken.sol
   Input test: 0x18B2f810B3592E4Cf4bC2DC0FB1317536e75dCC5
   Input main: 0x4C1598892E3EEEb9dE18CdaE7698545EC0680f7F

# 4.0 Management Instructions

## 4.1 CompTT

### 4.1.1 Pause Margin Token

## 4.2 MarginToken

### 4.2.1 Add a new token

1. _setFeed() in ChainlinkOracle.sol

WBNB test: 0xae13d989daC2f0dEbFf460aC112a837C89BAa7cd,0x2514895c72f50D8bd4B4F9b1110F0D6bD2c97526
BTCB test:  0xA808e341e8e723DC6BA0Bb5204Bafc2330d7B8e4,0x5741306c21795FdCBb9b265Ea0255F499DFe515C

WBNB main: 0xbb4CdB9CBd36B01bD1cBaEBF2De08d9173bc095c,0x0567F2323251f0Aab15c8dFb1967E4e8A7D42aeE
BTCB main:  0x7130d2A12B9BCbFAe4f2634d864A1Ee1Ce3Ead9c,0x264990fbd0A4796A3E3d8E37C4d5F87a3aCa5Ebf
ETH main:     0x2170Ed0880ac9A755fd29B2688956BD959F933F8,0x9ef1B8c0E4F7dc8bF5719Ea496883DC6401d5b2e
USDT main:  0x55d398326f99059fF775485246999027B3197955,0xB97Ad0E74fa7d920791E90258A6E2085088b4320
ADA main:    0x3EE2200Efb3400fAbB9AacF31297cBdD1d435D47,0xa767f745331D267c7751297D982b050c93985627
LINK main:    0xF8A0BF9cF54Bb92F17374d9e9A321E6a111a51bD,0xca236E327F629f9Fc2c30A4E95775EbF0B89fac8
XRP main:    0x1D2F0da169ceB9fC7B3144628dB156f3F6c60dBE,0x93A67D414896A280bF8FFB3b389fE3686E014fda
LTC main:     0x4338665CBB7B2485A8855A139b75D5e34AB0DB94,0x74E72F37A8c415c8f1a98Ed42E78Ff997435791D
DOT main:    0x7083609fCE4d1d8Dc0C979AAb8c869Ea2C873402,0xC333eb0086309a16aa7c8308DfD32c8BBA0a2592
UNI main:     0xBf5140A22578168FD562DCcF235E5D43A02ce9B1,0xb57f259E7C24e56a1dA00F66b55A5640d9f9E7e4
AVAX
NEAR
SOL
MATIC

## 2. _updateFeePerToken() in IncentiveModelSimple.sol

WBNB: 0xae13d989daC2f0dEbFf460aC112a837C89BAa7cd,1000000000000000
BTCB: 0xA808e341e8e723DC6BA0Bb5204Bafc2330d7B8e4,1000000000000000

WBNB main: 0xbb4CdB9CBd36B01bD1cBaEBF2De08d9173bc095c,1000000000000000
BTCB main: 0x7130d2A12B9BCbFAe4f2634d864A1Ee1Ce3Ead9c,1000000000000000
ETH main: 0x2170Ed0880ac9A755fd29B2688956BD959F933F8,1000000000000000
USDT main: 0x55d398326f99059fF775485246999027B3197955,500000000000000
ADA main: 0x3EE2200Efb3400fAbB9AacF31297cBdD1d435D47,1500000000000000
LINK main: 0xF8A0BF9cF54Bb92F17374d9e9A321E6a111a51bD,1500000000000000
XRP main: 0x1D2F0da169ceB9fC7B3144628dB156f3F6c60dBE,1500000000000000
LTC main: 0x4338665CBB7B2485A8855A139b75D5e34AB0DB94,1500000000000000
DOT main: 0x7083609fCE4d1d8Dc0C979AAb8c869Ea2C873402,1500000000000000
UNI main: 0xBf5140A22578168FD562DCcF235E5D43A02ce9B1,1500000000000000

## 3. _supportUnderlying() in CompTT.sol

WBNB: 0xae13d989daC2f0dEbFf460aC112a837C89BAa7cd
BTCB: 0xA808e341e8e723DC6BA0Bb5204Bafc2330d7B8e4

WBNB main: 0xbb4CdB9CBd36B01bD1cBaEBF2De08d9173bc095c
BTCB main: 0x7130d2A12B9BCbFAe4f2634d864A1Ee1Ce3Ead9c
ETH main: 0x2170Ed0880ac9A755fd29B2688956BD959F933F8
USDT main: 0x55d398326f99059fF775485246999027B3197955
ADA main: 0x3EE2200Efb3400fAbB9AacF31297cBdD1d435D47
LINK main: 0xF8A0BF9cF54Bb92F17374d9e9A321E6a111a51bD
XRP main: 0x1D2F0da169ceB9fC7B3144628dB156f3F6c60dBE
LTC main: 0x4338665CBB7B2485A8855A139b75D5e34AB0DB94
DOT main: 0x7083609fCE4d1d8Dc0C979AAb8c869Ea2C873402
UNI main: 0xBf5140A22578168FD562DCcF235E5D43A02ce9B1

## 4. _updateDepositsDisabled

Input test: [0xae13d989daC2f0dEbFf460aC112a837C89BAa7cd,0xA808e341e8e723DC6BA0Bb5204Bafc2330d7B8e4],true

Input main: [0xbb4CdB9CBd36B01bD1cBaEBF2De08d9173bc095c,0x55d398326f99059fF775485246999027B3197955],true
Input main ALL:
[0xbb4CdB9CBd36B01bD1cBaEBF2De08d9173bc095c,0x55d398326f99059fF775485246999027B3197955,0x7130d2A12B9BCbFAe4f2634d864A1Ee1Ce3Ead9c,0x2170Ed0880ac9A755fd29B2688956BD959F933F8,0x3EE2200Efb3400fAbB9AacF31297cBdD1d435D47,0xF8A0BF9cF54Bb92F17374d9e9A321E6a111a51bD,0x1D2F0da169ceB9fC7B3144628dB156f3F6c60dBE,0x4338665CBB7B2485A8855A139b75D5e34AB0DB94,0x7083609fCE4d1d8Dc0C979AAb8c869Ea2C873402,0xBf5140A22578168FD562DCcF235E5D43A02ce9B1],true

## 5. _updatePortfolioAndAllocations() in MarginTokens.sol

Input test:
[0xae13d989daC2f0dEbFf460aC112a837C89BAa7cd,0xA808e341e8e723DC6BA0Bb5204Bafc2330d7B8e4],[1000000000000000000,0],[0,0],[0,0]

Input main:
[0xbb4CdB9CBd36B01bD1cBaEBF2De08d9173bc095c,0x55d398326f99059fF775485246999027B3197955],[1000000000000000000,0],[0,0],[0,0]
Input Main:
[0xbb4CdB9CBd36B01bD1cBaEBF2De08d9173bc095c,0x55d398326f99059fF775485246999027B3197955,0x7130d2A12B9BCbFAe4f2634d864A1Ee1Ce3Ead9c,0x2170Ed0880ac9A755fd29B2688956BD959F933F8],[1000000000000000000,0,0,0],[0,0,0,0],[0,0,0,0]

Input Main 10:
[0xbb4CdB9CBd36B01bD1cBaEBF2De08d9173bc095c,0x55d398326f99059fF775485246999027B3197955,0x7130d2A12B9BCbFAe4f2634d864A1Ee1Ce3Ead9c,0x2170Ed0880ac9A755fd29B2688956BD959F933F8,0x3EE2200Efb3400fAbB9AacF31297cBdD1d435D47,0xF8A0BF9cF54Bb92F1737

4d9e9A321E6a111a51bD,0x1D2F0da169ceB9fC7B3144628dB156f3F6c60dBE,0x4338665CBB7B2485A8855A139b75D5e34AB0DB94,0x7083609fCE4d1d8Dc0C979AAb8c869Ea2C873402,0xBf5140A22578168FD562DCcF235E5D43A02ce9B1],[1000000000000000000,0,0,0,0,0,0,0,0,0],[0,0,0,0,0,0,0,0,0,0],[0,0,0,0,0,0,0,0,0,0]

6. _updateDepositsDisabled

Input test: [0xae13d989daC2f0dEbFf460aC112a837C89BAa7cd,0xA808e341e8e723DC6BA0Bb5204Bafc2330d7B8e4],false

Input main: [0xbb4CdB9CBd36B01bD1cBaEBF2De08d9173bc095c,0x55d398326f99059fF775485246999027B3197955],false

### 4.3.2 Support token supply and borrow to Venus

1. Steps 1-3 from 4.3.1
2. _supportVToken(token,vToken) in CompTT

   Input test: 0xae13d989daC2f0dEbFf460aC112a837C89BAa7cd,0x2E7222e51c0f6e98610A1543Aa3836E092CDe62c
   Input test: 0xA808e341e8e723DC6BA0Bb5204Bafc2330d7B8e4,0xb6e9322C49FD75a367Fcb17B0Fcd62C5070EbCBe

   BNB main: 0xbb4CdB9CBd36B01bD1cBaEBF2De08d9173bc095c,0xA07c5b74C9B40447a954e1466938b865b6BBea36
   USDT main: 0x55d398326f99059fF775485246999027B3197955,0xfD5840Cd36d94D7229439859C0112a4185BC0255
   Input Main: 0x7130d2A12B9BCbFAe4f2634d864A1Ee1Ce3Ead9c,0x882C173bC7Ff3b7786CA16dfeD3DFFfb9Ee7847B

3. _updateTrendTokenVenusActions() in CompTT
   a. set to true for supply and borrow as desired

4. _updateUnderlyingForVenusAction() in CompTT
   a. Set to true for isVenus, isSupply, and isBorrow as desired

5. _updatePortfolioAndAllocations() in MarginTokens.sol
   a. Input: [tokens],[contract],[>0 collateral],[>0 borrow]

### 4.3.3 Removing a token

1. _setDesiredAllocations() to [0],[0],[0] for token desired to be removed
2. _depositsDisabled(token,true) for token to be removed
3. systematically repay, redeem, and remove asset from pool
   a. Make sure contract+collateral+borrow is below maxDisableValue
4. _updatePortfolioAndAllocations() to not include removed token

- To remove a token, it is desired that it is disabled from collateral. In order for a token to be disabled from Venus it must not:
    - Have a current borrow balance
    - Have its collateral rely on a current borrow balance (even in another token)

# 5.0 Test Cases

Below are some cases to test. The list is not exhaustive.

## 5.1 CompTT

### 5.1.1 Admin

`_supportVToken():`

- Support true wbnb/vBNB: success, added
- Support wbnb/vBNB again: success, unable to add already supported
- Support false btcb/vbtcb: success, unable to add
- Support true btcb/vbtcb: success, able to add

`_updateTrendTokenActiveStatus():`

- Unpause Trend Token: success, unpaused and able to deposit, redeem, etc

`_updateUnderlyingActiveStatus():`

- Unpause underlying: success, unpaused and able to deposit, etc

### 5.1.2 Pause Guardian

`_setProtocolPaused():`

`_updateTrendTokenActiveStatus():`

- Set Trend Token to Inactive: success, unable to deposit, redeem, etc
- Unpause: success, only admin can

`_updateUnderlyingActiveStatus():`

- Set USDT to Inactive: success, unable to deposit
- Unpause, success, only admin can

## 5.2 MarginTokens

### 5.2.1 User Actions

`depositBNB():`

- Deposit 0.01: success, able to deposit
- depositDisabled[BNB]: success, unable to deposit
- trendTokenPaused: success, unable to deposit
- Set low maxTradeFee and trade: success, unable to trade → !maxTradeFee

`deposit():`

- Deposit 0.01: success, able to deposit
- Deposit 1: success, unable to deposit → `maxSupply` and `maxTradeValue`
- depositDisabled[BTCB]: success, unable to deposit
- trendTokenPaused: success, unable to deposit
- Deposit with minOut too low: success, unable to deposit → !minOut
- Set low maxTradeFee and trade: success, unable to trade → !maxTradeFee

`redeem():`

- Redeem 0.01 BTCB: success, able to redeem
- Redeem 0.01 BNB: success, able to redeem
- Redeem 1 BTCB: success, unable to redeem → `maxTradeValue`
- Redeem 10 BTCB: success, unable to redeem → `maxTradeValue` and `insufficient`
- Set low maxTradeFee and trade: success, unable to trade → !maxTradeFee
-

`swapExactBNBForTokens():`

- 0.01 BNB for BTCB: success, approx 0.000088
- trendTokenPaused: success, unable to trade
- Set low maxTradeFee and trade: success, unable to trade → !maxTradeFee

`swapExactTokensForTokens():`

- 0.000088 BTCB for BNB: success, approx 0.00995 BNB
- trendTokenPaused: success, unable to trade
- Set low maxTradeFee and trade: success, unable to trade → !maxTradeFee

`executeSupply():`

- When venusOpen is false: success, unable to supply
- When venusOpen is true, success, able to supply

### 5.1.2 Systems Tests:

- Borrow BTCB to go long BNB: success, short BTCB to leverage long BNB
- Borrow BNB to go long BTCB: success, short BNB to leverage long BTCB
- Borrow and try to redeem from Venus beyond liquidation

### 5.2.2 tradingBot Actions

`_updateVenusAndPauseState()`:

- openVenus to true: success, public able to interact with Venus
- Pause to true:  success, unable to do user or venus interactions

`_updateDepositsDisabled()`:

- Disable BNB and BTCB: success, unable to supply token and can update positions
- Enable BNB and BTCB: success, able to supply and update positions

`_updatePortfolioAndAllocations()`:

- Without `depsoitsDisabled[]`: success, unable to update
- Add supply without supporting vToken: success, unable to add
- Add supply after supporting vToken: success, able to add
- Collateral BNB borrow BTCB: success, able to update
- Collateral BTCB borrow BNB: success, able to update
- Remove a token with value above maxDisableTokenValue: success, unable
- Remove a token below maxDisableTokenValue: success, able to

`executeSupply()`:

- Supply when desired: success, able to supply
- Redeem when not desired: success, unable to → supplyDirectionCheck
- Redeem more than supplied: success, unable to redeem → exceeded supply
- Redeem when desired: success, able to
- Supply when not desired: success, unable to → supplyDirectionCheck

`executeBorrow()`:

- Borrow when desired: success, able to borrow
- Borrow when not desired: success, unable to → borrowDirectionCheck
- Repay more than borrowed: success, unable to redeem → exceeded supply
- Repay BNB when desired: success, able to
- Repay BTCB when desired: success, able to
- Repay when not desired: success, unable to → borrowDirectionCheck

### 5.2.3 manager Actions

- Redeem fees
- Update variables
- Adjust performance fees

`_updateCompAndIncentive():`

- Update compTT: success, updated
- Update Incentive: success, updated

`_updatePerformanceFee():`

- Set 80% performance fee: success, unable to
- Set 5% performanceFee: success, able to

`_updateMaxDisableAndSupply():`

- Set 100000 maxDisableValue:
- Set 1B maxSupply:

`_claimXVSandFees():`

- Try to redeem more XVS and Margin Tokens than in contract

## 5.3 Accessory Files

### 5.3.1 ChainlinkOracle

- Add token not 18 decimals: success, unable to add
- Add token 18 decimals: success, able to add
- Call token from underlying: success, able to get price

### 5.3.2 IncentiveModel

- Change feePerToken: success, changed
- Ensure fees with deposits, redeems, swaps, with Margin Tokens make sense: success

# 6.0 Potential Exploits and Prevention

There are a variety of potential exploits and protective measures in place to prevent such situations. They can be broken up into 2 main categories:

1) User Interactions - buy, sell, swap
2) Venus Interactions - supply, redeem, borrow, swap

## 6.1 User Interactions

The public has a few contracts available to them, including functions to deposit assets to buy Margin Tokens (`deposit()` and `depositBNB()`), redeem assets from Margin Tokens (`redeem()`), and swap one asset for another using the Margin Token Portfolio (`swapExactTokensForTokens()` and `swapExactETHForTokens()`).

The price of a Margin Token is calculated as follows:

*Margin Token Price = (value in portfolio) / (supply of Margin Tokens)*

There are a variety of ways users could affect the price of Margin Tokens which will be described below:

### 6.1 Manual Actions (increase price)

#### 6.1.1 Manually Deposit Assets

A user could manually send an asset into the pool without calling the `deposit()` or `depositBNB()`function. This would increase the ‘value in portfolio’ without increasing ‘supply of Margin Tokens’, thereby increasing the price.

Protection

There is no protection for this and not necessary as it has no benefit to users and only benefits Margin Token Holders through increased price appreciation

#### 6.1.2 Manually Buring Margin Tokens

A user could burn Margin Tokens by calling the `burn()` function from the Margin Token token contract. This would reduce the ‘supply of Margin Tokens’ without affecting the ‘value in portfolio’ and therefore increase the price of Margin Tokens

Protection

There is no protection for this and not necessary as it has no benefit to users and only benefits Margin Token Holders through increased price appreciation

### 6.2 Sandwich Attack (decrease price)

Let’s assume the portfolio desires 100% in USDT and the portfolio currently holds 99% in USDT. In this state, there is a reward (assume 0.1%) for depositing USDT into the pool as it brings the current USDT positions (99%) closer to the desired positions (100%).

As the market switches from downtrend to uptrend, the tradingBot sends a transaction to change the desired positions from 100% USDT to 0% USDT, thereby offering a 0.10% reward to users for buying USDT as it brings the current positions (99%) towards the desired positions (0%).

A front-running bot can perform a sandwich attack by posting a transaction just before the update and just after the tradingBot update.

Upon the attacker detecting the `tradingBot` transaction to switch desired positions from 100% USDT to 0% in the mempool, he can:

1) Take out a massive USDT flash loan (100 million) and deposit it into the Margin Token Portfolio for Margin Tokens to earn the 0.1% fee ($100,000). This is done at a high gas fee to ensure it is earlier in the block than the `tradingBot` desired allocation update.

2) Allow the tradingBot to change desired positions from 100% USDT to 0% USDT

3) Send another transaction to redeem Margin Tokens from step 1 for USDT, earning another 0.10% fee ($100,000)

4) Repay the initial flash loan and pocket the profit

The attacker effectively earned a $200,000 profit from exploiting the reward mechanism.

A similar attack could occur with the `swapExactTokensForTokens()` or `swapExactETHForTokens()` functions but to a lesser degree since the scale would be limited by the assets in the pool, rather than the infinite supply of minting new Margin Tokens and in practice only help contribute to the rebalance of the pool which is desired.

### Protection

In order to protect against this sandwich attack, two checks were put in place.

1) There is a Margin Token `maxSupply` that limits the amount of total Margin Tokens that can be in supply. At the end of the `deposit()` and `depositBNB()` functions, this is checked and the transaction is reverted if newly minted Margin Tokens upon deposit would exceed this Margin Token supply. Although this protection goes a long way in preventing large scale attacks, there

could still be repeated small scale attacks of up to 0.25% the difference between current Margin Token supply and `maxSupply`.

2) Before the `tradingBot` can update the positions with `_setDesiredAllocationsFresh()`, all deposits of each token in the current portfolio must be disabled. This should be done in a block that is not the same as the one in which the positions will be updated. This will provide a second layer of protection beyond the first in order to prevent the sandwich attack. The check for this is done by `compTT.permissionPortfolio()`

### 6.3 Large Redeem

If a single person holds a large share of the pool (99%+) they could substantially decrease the price of Trend Tokens upon redeeming their share, as their reward (or fee) would be large relative to the remaining assets in the pool.

Let's assume User A owns the vast majority of a $100,000 Margin Token Portfolio and there are 100,000 Margin Tokens in circulation.

*Margin Token Price = $100,000 in portfolio / 100,000 margin tokens in circulation = $1*

| **Users** | **Share of Pool** | **Margin Tokens** | **Value** |
|---|---|---|---|
| User A | 99% | 99,000 | $99,000 |
| All other users | 1% | 1,000 | $1,000 |

Lets assume the User A redeems his 99,000 Margin Tokens for a reward of 0.10%.

*Reward = $99,000 sell in Margin Tokens * 0.10%*
*= $99 reward*

*Value Out = (99,000 + 99) * $1 per Margin Token*
*= $99,099*

After User A redeems 99,000 Margin Tokens and receives $99,099 after the incentive reward, there will only be $901 in the pool

*Margin Token Price = $901 in pool / 1000 Margin Tokens in circulation = $0.901*

Therefore, User A effectively reduced the value of Margin Tokens by redeeming the vast majority of Trend Tokens. A similar result would be achieved even if there was a fee instead of a reward. In this case, there would be $1000 in the pool but an over-supply of Margin Tokens due to the fee remaining in the contract for the manager to redeem at any time.

#### Protection

Although this would happen in only rare and extreme cases when a whale has acquired a vast majority of available Margin Tokens, it is still unfair to the remaining Margin Token holders if this were to happen.

1) A value of `maxTradeValues` is set in CompTT to limit the value of deposit, redeem, or trade when interacting with a Margin Token Portfolio. If a large holder wishes to sell their stake, they will need to do this in several transactions and incur negative price changes as they accomplish the trades. CompTT will only grant permission for the above listed actions if the trade value is less than `maxTradeValues`. Furthermore, this may help limit or reduce any large scale attack vectors which often are amplified with the use of flash loans.

## 6.1 Venus Interactions

The `tradingBot` has permission to use the Margin Token Portfolio assets to supply, redeem, borrow, and repay with Venus. If `venusOpen` is true, then the public may execute these actions as well.

### 6.1.1 Liquidation

When borrowing assets from Venus, after supplying collateral, there is a risk of liquidation if your collateral value falls below a threshold. There is a risk the tradingBot, or the public, may intentionally or unintentionally borrow too much relative to the supplied collateral and risk liquidation and the heavy fees that apply in these cases.

In order to reduce this risk, a variety of measures were put in place, including:

1) Limitations are set for `_setDesiredAllocationsFresh`() when the `tradingBot` updates desired `contract`, `collateral`, and `borrow` allocations. The `tradingBot` cannot set `borrow` allocations that would exceed `maxBorrowFactor`. The `maxBorrowFactor` is the maximum amount the Margin Token Portfolio can borrow relative to the total amount that can be borrowed based on the supplied collateral. For example, if $100,000 is supplied to Venus with an average collateral factor of 80%, then a total of $80,000 is borrowable from Venus. A `maxBorrowFactor` of 50% means that

the Margin Token Portfolio has permission to borrow up to $40,000. In this case, the collateral would need to fall 50% before the Margin Token Portfolio is liquidated.

2) Whenever Margin Token Portfolio assets are redeemed from Venus or assets are borrowed, the borrow factor after this action is checked and compared to the `maxBorrowFactor` to ensure it is not exceeded. This is done in `CompTT` by the `performBorrowFactorCheck()`.
3) In order for interaction with Venus to occur, the desired action is compared to the current allocations and desired allocations as set by the `tradingBot` to ensure the desired action brings the current closer to the desired. This is done in `CompTT` by the `performSupplyDirectionCheck()` and `performBorrowDirectionCheck()`.