Assure DeFi®

THE VERIFICATION GOLD STANDARD

Security Assessment

OnlyPump - Fee manager

Date: 15/10/2025

Audit Status: PASS

Audit Edition: Advanced+

ASSURE DEFI®

THE VERIFICATION GOLD STANDARD

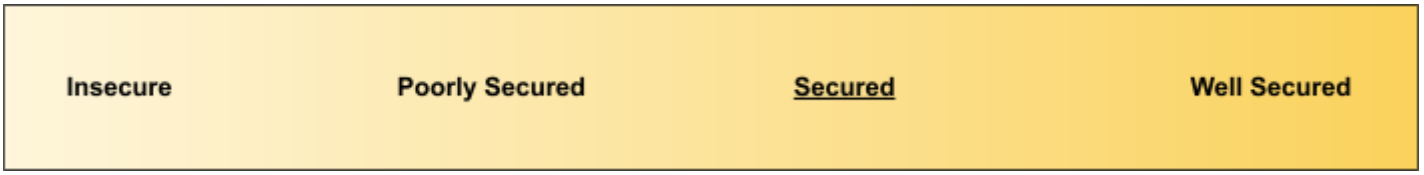
Risk Analysis

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Low	Low-level vulnerabilities are primarily associated with outdated or unused code snippets that generally do not significantly impact execution, sometimes they can be ignored.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 **Secured.**



Scope

Target Code And Revision

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

Target Code And Revision

Project	Assure
Language	Solidity
Codebase	Token.sol [SHA256]: 9c3a15bc662c86d1ca99822b340fbc52852f3a290f94f21d51bab8c38605f9ef Master.sol [SHA256]: b2a94f51d8ea2e5774699918 9a35c71c28c7502eed6f072f7180bbfd9d7f2680 FeeManager.sol [SHA256] 793e10771aa02445bfa169596b41c25deec07a6fbb2401963717760e0ae9d6da
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

No high severity issues were found.

MEDIUM

1. Integer-division “dust” in investor distribution causes silent accounting drift (funds become untracked until future inflows)

Location: FeeManager.distributPlatofrmFees()

Issue: Because each walletShare uses integer division, the sum of per-wallet floors can be less than totalToDistribute by up to (walletCount - 1) wei per distribution (worse with many wallets and small distributions). totalPlatformFees is set to 0 before allocations, and the remainder is not carried forward or recorded anywhere. The leftover wei stays in the contract balance but is not reflected in any accounting mapping and is not directly withdrawable by anyone.

Recommendation: Implement a remainder carry strategy:

Accumulate a running distributionRemainder and add it to the next distribution before the per-wallet split or

Compute all but the last wallet with integer division and assign the entire remaining amount to the final wallet to guarantee exact conservation.

Alternatively, keep totalPlatformFees intact until the sum of allocations is computed, subtract exactly that sum, carry the difference forward (in totalPlatformFees), and emit it as dustCarried for transparency.

2. No slippage bounds in buy()/sell()

Location: Token.SimulateBuy()

Issue: Potential mempool front-running / sandwich risk callers use simulateBuy() off-chain to estimate cost, but buy()/sell() accept trades without caller-controlled slippage bounds or deadlines. An MEV bot can observe a pending tx, front-run or sandwich it, move the price, and make the user receive far fewer tokens (or far less ETH on sells)

Recommendation: Add explicit slippage protection and execution window parameters and enforce them on-chain. Minimal changes: buy(uint256 minTokensOut, uint256 deadline) — require block.timestamp <= deadline and tokensOut >= minTokensOut where tokensOut is computed with the exact same logic as _simulateBuy(). sell(uint256 amountIn, uint256 minEthOut, uint256 deadline) — same pattern for sells.

LOW

1. FeeManager is fragile to misconfigured or mismatched Master; no explicit compatibility assertion

Location: FeeManager.setMaster() and FeeManager.distributePlatformFees()

Issue: Master.getProfitWallets/Percentages require msg.sender == feeManager (an immutable address set in Master's constructor). If FeeManager.owner mistakenly calls setMaster with a Master whose feeManager is a different address, distributePlatformFees() will revert forever, bricking investor distributions (though creator withdrawals still work). The setMaster() one-way lock prevents correction.

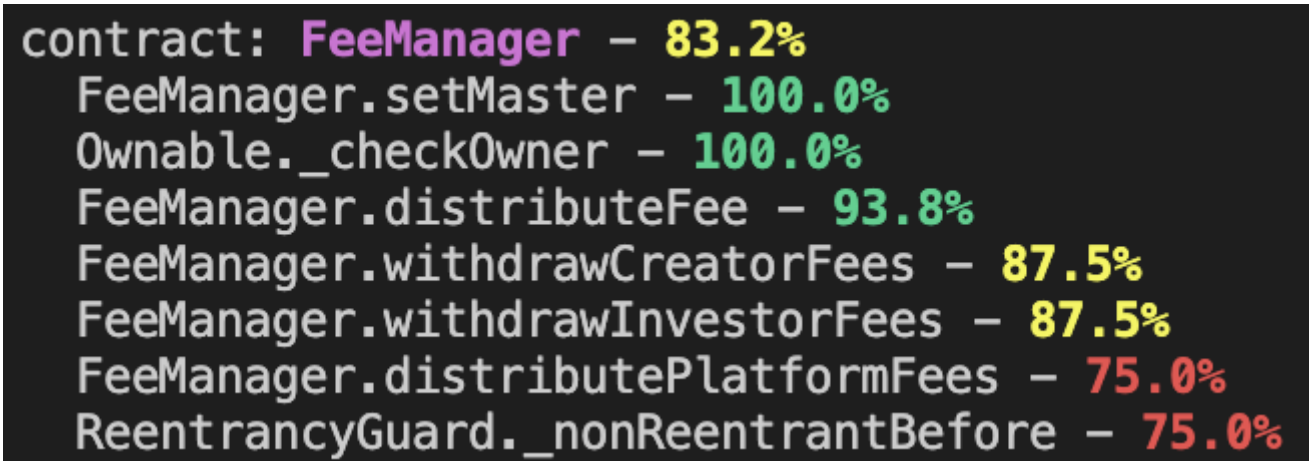
Recommendation: In setMaster assert compatibility: Add require(IMaster(_master).getProfitWalletCount() > 0, "Bad master"); and a read-only handshake (e.g., IMaster(_master).feeManager() == address(this)) by exposing a public getter in Master for feeManager.

INFORMATIONAL

No informational issues were found.

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.*



```
tests/test_fee_manager.py::test_fee_manager_distribution_and_withdraw RUNNING
Transaction sent: 0x13ff6a351a7deb147a46e61aca43b90ac6a0585519e1516fb2d7bcc030c99a1c
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 0
  WETH9.constructor confirmed   Block: 1   Gas used: 476546 (3.97%)
  WETH9 deployed at: 0x3194cBDC3dbcd3E11a07892e7bA5c3394048Cc87

Transaction sent: 0x06e49bd474f045f693aed487b14efd01fdf94add86a82239d27818895cb94b81
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 1
  UniswapV2Router02.constructor confirmed   Block: 2   Gas used: 3895430 (32.46%)
  UniswapV2Router02 deployed at: 0x602C71e4DAC47a042Ee7f46E0aee17F94A3bA0B6

Transaction sent: 0xe3118af866bbe8d43e8a92ddebc3970e6054fc3fbb258fb64ee323b1d91ff95c
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 2
  FeeManager.constructor confirmed   Block: 3   Gas used: 832596 (6.94%)
  FeeManager deployed at: 0xE7eD6747FaC5360f88a2EFC03E00d25789F69291

Transaction sent: 0x1b1199f3302a0df8673dcb4ae45da3257f06e81cc1016d3fe3a914a71d57a7c1
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 3
  Master.constructor confirmed   Block: 4   Gas used: 3040801 (25.34%)
  Master deployed at: 0x6951b5Bd815043E3F842c1b026b0Fa888Cc2DD85

Transaction sent: 0xd3db0fd55812b671a5d2adba3ba233af8779096a7e4892e6cc0ddf2fc828eddc
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 4
  FeeManager.setMaster confirmed   Block: 5   Gas used: 44416 (0.37%)

Transaction sent: 0xe37f140722171199fda4bdc2645a30f9590f184ebe2cbb99978db9febba78511
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 0
  Token.constructor confirmed   Block: 6   Gas used: 1873227 (15.61%)
  Token deployed at: 0xe7CB1c67752cBb975a56815Af242ce2Ce63d3113

Transaction sent: 0x78ab1334a8c7db02dafc4abf476a0563d7beae46d4cc3fafd19a5ebe157f8b8f
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 1
  Master.deploy confirmed   Block: 7   Gas used: 1921319 (16.01%)

Transaction sent: 0xa516410b70d799a0bd7416e91ba09710e522e5da60afe0b3d1d4e732dad0dd7c
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 2
  Transaction confirmed   Block: 8   Gas used: 82002 (0.68%)

Transaction sent: 0xb8af495edf3c8c9774a2042524c2fb845a5f2b2a8d070ccee03cae25532c8294
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 0
  FeeManager.withdrawCreatorFees confirmed (reverted)   Block: 9   Gas used: 28065 (0.23%)

Transaction sent: 0x8d32f1e42651c519bf5338583203d06a1e1e2d178b520bbd87d75704f4768ec0
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 3
  FeeManager.withdrawCreatorFees confirmed   Block: 10   Gas used: 23685 (0.20%)

Transaction sent: 0x28c4e77d4c97e99cda6389a4d8d49f0f765855b661f82c341c3533a95a06f952
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 1
  FeeManager.distributePlatformFees confirmed   Block: 11   Gas used: 95769 (0.80%)

Transaction sent: 0x2171947d6090af9dcdd735203b4ef26b155331b12fa1e615e7e692fce6e64e3f
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 2
  FeeManager.withdrawInvestorFees confirmed (reverted)   Block: 12   Gas used: 28022 (0.23%)

Transaction sent: 0x7f1b0ee76c936ca79b1f11508e863c88d0443c603b9d5b1689301e4687a9932b
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 0
  FeeManager.withdrawInvestorFees confirmed   Block: 13   Gas used: 23631 (0.20%)

tests/test_fee_manager.py::test_fee_manager_distribution_and_withdraw PASSED
```

```
tests/test_fee_manager.py::test_set_master_and_failures RUNNING
Transaction sent: 0x55e4c25657c2e80b6e2135baeeb83efe7fcf1fda133cbf1adee16cdec2537191
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 5
  FeeManager.constructor confirmed   Block: 14   Gas used: 832596 (6.94%)
  FeeManager deployed at: 0x6b4BDe1086912A6Cb24ce3dB43b3466e6c72AFd3

Transaction sent: 0x87fdff20583866af95bb34e36783d52eb00806c53f1ab4e57aa8f068d0210732
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 4
  FeeManager.setMaster confirmed (reverted)   Block: 15   Gas used: 22721 (0.19%)

Transaction sent: 0xd9de2ce3bac56c01d71324f3de10e57950b4e37a6ca3cd8665d877910d1f61fe
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 6
  FeeManager.setMaster confirmed (reverted)   Block: 16   Gas used: 23339 (0.19%)

Transaction sent: 0xf6ee9ae1aabfa8ce25963fc091100c7b974011e5a02ded564f7ee57d94ecb79e
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 7
  FeeManager.setMaster confirmed   Block: 17   Gas used: 44416 (0.37%)

Transaction sent: 0xe304cf69fc23432a93c5fd496729c3e08e7cb2fa31685e05189eaf785d3fcc1d
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 8
  FeeManager.setMaster confirmed (reverted)   Block: 18   Gas used: 23544 (0.20%)

tests/test_fee_manager.py::test_set_master_and_failures PASSED
tests/test_fee_manager.py::test_distribute_fee_validations RUNNING
Transaction sent: 0x1f3d86d45d8d1a24ee558f2e8f674b4943752a046b085204ce54d461a024161a
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 9
  FeeManager.constructor confirmed   Block: 19   Gas used: 832596 (6.94%)
  FeeManager deployed at: 0xa3B53dDCd2E3fC28e8E130288F2aBD8d5EE37472

Transaction sent: 0xd7951a0baeea6431ada49f700df79228af13e2e168fbcc0319bf00f432c87813
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 10
  Master.constructor confirmed   Block: 20   Gas used: 3040825 (25.34%)
  Master deployed at: 0xb6286fAFd0451320ad6A8143089b216C2152c025

Transaction sent: 0x92a4e0e043fcbf1e533cc605109846b6579c907e1165487ba6b64e1a1c7db76a
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 11
  FeeManager.setMaster confirmed   Block: 21   Gas used: 44416 (0.37%)

Transaction sent: 0x66637e54ad9e9d660cff2761c2eb28a4656f9791b26c1a7f61159aafc685ace9
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 12
  WETH9.constructor confirmed   Block: 22   Gas used: 476546 (3.97%)
  WETH9 deployed at: 0x2c15A315610Bfa5248E4CbCbd693320e9D8E03Cc

Transaction sent: 0x603af0de7f64fb280dafdbcd7379131509cf8d260010c032f8c6e160efc58778
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 13
  UniswapV2Router02.constructor confirmed   Block: 23   Gas used: 3895190 (32.46%)
  UniswapV2Router02 deployed at: 0xe692Cf21B12e0B2717C4bF647F9768Fa58861c8b

Transaction sent: 0xd4dcd58a5c414a2b6a3989b5bbf4d2044ed3994a9be3872387ddf4be4c5d0e89
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 5
  Token.constructor confirmed   Block: 24   Gas used: 1873227 (15.61%)
  Token deployed at: 0x8F37Fb31d618513553fdF93e90c4C11BD8bf112c

Transaction sent: 0x7ce6d53ec00931160f1d585df8f803eb48556e44d9ad4d63bb64ae92d24e3fec
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 6
  FeeManager.distributeFee confirmed (reverted)   Block: 25   Gas used: 22687 (0.19%)

Transaction sent: 0xdefa3787a911c2e6c075561b71172166eb54ee82a2c863219bdf6062350ee10b
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 14
  FeeManager.constructor confirmed   Block: 26   Gas used: 832596 (6.94%)
  FeeManager deployed at: 0xe65A7a341978d59d40d30FC23F5014FACB4f575A

Transaction sent: 0xfccc9cc8a9983d43c2ba92f04f4ed71c5048c47ae680add6198a5fd2db015066
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 7
```

```
tests/test_token_master.py::test_buy RUNNING
Transaction sent: 0xdb7a1c4a1bc96a251c59761c7913be53c85ed42e1c3202623f9cc29a83754a57
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 15
  UniswapV2Factory.constructor confirmed   Block: 29   Gas used: 2412742 (20.11%)
  UniswapV2Factory deployed at: 0x30375B532345B01cB8c2AD12541b09E9Aa53A93d

Transaction sent: 0xe9838c712dfcb93e3d1be7eca832d448b46cfa5aff484799970cd36f952101da
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 16
  WETH9.constructor confirmed   Block: 30   Gas used: 476546 (3.97%)
  WETH9 deployed at: 0x26f15335BB1C6a4C0B660eDd694a0555A9F1cce3

Transaction sent: 0xdfb74272825d09eb3d8bb4ef9a596419b1899850cacdf3bda7704c9b1f46412d
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 17
  UniswapV2Router02.constructor confirmed   Block: 31   Gas used: 3895430 (32.46%)
  UniswapV2Router02 deployed at: 0xFbD588c72B438faD4Cf7cD879c8F730Faa213Da0

Transaction sent: 0x2c971fc996187c3a4588f14ae949c53f6caea3b26c5bb5d14cbff849fb2bb893
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 18
  FeeManager.constructor confirmed   Block: 32   Gas used: 832596 (6.94%)
  FeeManager deployed at: 0xed00238F9A0F7b4d93842033cdF56cCB32C781c2

Transaction sent: 0x22b87ba5c37011b7abf1311f66b526794fe1c5dbbb73ee04f404dd843a9bd3b3
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 19
  Master.constructor confirmed   Block: 33   Gas used: 3040813 (25.34%)
  Master deployed at: 0xDae02e4fE488952cFB8c95177154D188647a0146

Transaction sent: 0x29abd4560993debc528bec690b907325b3648138779ce72d5bb5dd5b6c41d93f
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 20
  FeeManager.setMaster confirmed   Block: 34   Gas used: 44416 (0.37%)

Transaction sent: 0xf2ab79365682181e6f1402a4b806243e2c26a68e98ac84003b900bc8c48cf0c3
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 21
  Token.constructor confirmed   Block: 35   Gas used: 1873227 (15.61%)
  Token deployed at: 0xBcb61491F1859f53438918F1A5aFCA542Af9D397

Transaction sent: 0x39fed2a5abd7385e8e997bd9c60e3915c68c601ec174f6ea6229821ae03336bc
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 9
  Master.deploy confirmed   Block: 36   Gas used: 1921319 (16.01%)

Transaction sent: 0x125f1b95e08c1c637ebf882ee5bdf6078c3cdf51f41aeb94b2825b2972a073a4
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 1
  Transaction confirmed (reverted)   Block: 37   Gas used: 33097 (0.28%)

Transaction sent: 0x423b36b0b702db8f10b7e5ee8dc8405b791f314dd0dd152f5a520bc6690733a5
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 2
  Transaction confirmed   Block: 38   Gas used: 82002 (0.68%)

Transaction sent: 0x1598f14b4438be8b733959702e941d87b9a88e7c9cf75f8261cd719c46a7f8e0
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 10
  Transaction confirmed   Block: 39   Gas used: 96990 (0.81%)

Transaction sent: 0x66d1e97fb90cb3068959034f2118399b01d515129170f519af0ae11e6055f827
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 3
  Transaction confirmed   Block: 40   Gas used: 96990 (0.81%)

Transaction sent: 0x16c7de850eda45ce142d2edb465b1b89f7b644044d8138bed16a62928ccaa5e4
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 11
  Transaction confirmed (reverted)   Block: 41   Gas used: 22178 (0.18%)

Transaction sent: 0x6607ebced02c2418d2685779617cdbe118f6ebb68abc7a95cd335fd15fec50a1
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 4
  Transaction confirmed   Block: 42   Gas used: 2327140 (19.39%)

Transaction sent: 0xcd3f5cd9e7c3201ec58b02c2a7efac91c2f860b580e36001f136c26bb7c23911
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 5
```

```
tests/test_token_master.py::test_check_buy RUNNING
Transaction sent: 0x1278a4eb0961e8ebb29f3c4128479341041cef054b72e2bb93f14d796ffa7b57
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 22
  UniswapV2Factory.constructor confirmed   Block: 46   Gas used: 2412742 (20.11%)
  UniswapV2Factory deployed at: 0xD22363efee93190f82b52FCD62B7Dbcb920eF658

Transaction sent: 0x9d0778a54bc1a694ed472bc42c9941bf80e46a0cf89830c2d8d75c0b60168ebb
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 23
  WETH9.constructor confirmed   Block: 47   Gas used: 476546 (3.97%)
  WETH9 deployed at: 0x4D1B781ce59B8C184F63B99D39d6719A522f46B5

Transaction sent: 0x38cb58af3504c138f8396c1530caec0046285835d74ff89e4648740f0e7330ad
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 24
  UniswapV2Router02.constructor confirmed   Block: 48   Gas used: 3895430 (32.46%)
  UniswapV2Router02 deployed at: 0xf9C8Cf55f2E520B08d869df7bc76aa3d3ddDF913

Transaction sent: 0xe6e001ce8ca25bb1b200d45baa99ef40c7c2fb3c7267342f406797c67b82f383
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 25
  FeeManager.constructor confirmed   Block: 49   Gas used: 832596 (6.94%)
  FeeManager deployed at: 0x654f70d8442EA18904FA1AD79114f7250F7E9336

Transaction sent: 0x0115b4173641706a13dac3d2fe66f4719b94cf52903eae89bd8beba0b9633cba
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 26
  Master.constructor confirmed   Block: 50   Gas used: 3040825 (25.34%)
  Master deployed at: 0xADeD61D42dE86f9058386D1D0d739d20C7eAfC43

Transaction sent: 0x022baead361ee0cc52751dea1ad37f5c82cca65313925ddc77d608d89b2e3704
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 27
  FeeManager.setMaster confirmed   Block: 51   Gas used: 44416 (0.37%)

Transaction sent: 0xf95a61b95c3c2760643b2c2e4082d66793e77eb651d981bc42b179b591719117
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 28
  Token.constructor confirmed   Block: 52   Gas used: 1873227 (15.61%)
  Token deployed at: 0xA95916C3D979400C7443961330b3092510a229Ba

Transaction sent: 0x6dc420890cced5672d9200f16d08058748e4ab4ab0abddad2399f59a8b5373f0
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 14
  Master.deploy confirmed   Block: 53   Gas used: 1921319 (16.01%)

Transaction sent: 0x0785baf089986c221b91debe4cf5f366972fae6597dd6dabb494b24f2b623492
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 3
  Transaction confirmed   Block: 54   Gas used: 96990 (0.81%)

tests/test_token_master.py::test_check_buy PASSED
```

```
tests/test_token_master.py::test_sell RUNNING
Transaction sent: 0x3a6aabfa56b7294feac01b690d5ff8c43aa596c31b6daa29d63420254013887b
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 29
  UniswapV2Factory.constructor confirmed   Block: 55   Gas used: 2412742 (20.11%)
  UniswapV2Factory deployed at: 0x42E8D004c84E6B5Bad559D3b5CE7947AADb9E0bc

Transaction sent: 0xe9e9ba64876d9f0430ea89e5446c6abcbf15d5d7d9d5f7f44f71d72cf492b030
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 30
  WETH9.constructor confirmed   Block: 56   Gas used: 476546 (3.97%)
  WETH9 deployed at: 0xF06D5f5BfFFCB6a52c84cfebc03AD35637728E73

Transaction sent: 0x52914dbdeb21f0c66223a6f690a2acf6c0f2e0829535c392b97e2c2925fa44cd
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 31
  UniswapV2Router02.constructor confirmed   Block: 57   Gas used: 3895430 (32.46%)
  UniswapV2Router02 deployed at: 0x82c83b7f88aef2eD99d4869D547b6ED28e69C8df

Transaction sent: 0xf1bb883e2ddf466dd7135ef61a458fff7f73818b5ae3166e923cba7a733165b5
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 32
  FeeManager.constructor confirmed   Block: 58   Gas used: 832596 (6.94%)
  FeeManager deployed at: 0x724Ca58E1e6e64BFB1E15d7Eec0fe1E5f581c7bD

Transaction sent: 0x0db42d26f9f56ab7afb835b6938007f110ce06e4ecf98365b51924cd41af0b15
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 33
  Master.constructor confirmed   Block: 59   Gas used: 3040825 (25.34%)
  Master deployed at: 0x34b97ffa01dc0DC959c5f1176273D0de3be914C1

Transaction sent: 0xcc9d3725a8b7184742078eb295b0987e3ee42622496e6a09e893a981194661a0
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 34
  FeeManager.setMaster confirmed   Block: 60   Gas used: 44416 (0.37%)

Transaction sent: 0x95d482c84cfa4c849d72268bc73dbce83903de1bab590720bed8c7aa8f8f66ce
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 35
  Token.constructor confirmed   Block: 61   Gas used: 1873227 (15.61%)
  Token deployed at: 0xbc8eCccb89650c3E796e803CB009BF9b898CB359

Transaction sent: 0x44879c297bcb6bdf8653832f88f598fff451a2d73c0a9e9438fdd0c7dc70ca8b
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 15
  Master.deploy confirmed   Block: 62   Gas used: 1921319 (16.01%)

Transaction sent: 0xfdc2fbca3400bc007872eae3b9afd4499ef47258a657431eb472bf86439b296a
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 4
  Transaction confirmed (reverted)   Block: 63   Gas used: 29143 (0.24%)

Transaction sent: 0xf910062f06e0d6be6d227239553f24727ff6dd89868b7136530d1f778beedd3d
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 5
  Transaction confirmed   Block: 64   Gas used: 96990 (0.81%)

Transaction sent: 0x99c5eae2a1a435bed704fef8ddd1445a4fd6e0d104ece34d0f7066c09f8f318b
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 6
  Transaction confirmed   Block: 65   Gas used: 87709 (0.73%)

tests/test_token_master.py::test_sell PASSED
```

Annexes

Testing code:

Fee Manager:

```
from brownie import reverts, Token


from brownie.network.contract import Contract


from scripts.helpful_scripts import (
    ZERO_ADDRESS,
    get_account,
    get_buy_fee,
    get_random_profit
)


from scripts.deploy import (
    deploy_master,
    deploy_fee_manager,
    deploy_token,
    deploy_weth,
    deploy_router,
)



def test_fee_manager_distribution_and_withdraw(only_local):
    # Arrange
    owner = get_account(0)
    creator = get_account(1)
    investor = get_account(2)
    someone = get_account(3)

    weth = deploy_weth(owner)
    router = deploy_router(owner, get_account(4), weth.address)
```

```
    fee_manager = deploy_fee_manager(owner)
    profit_wallets, profit_perf = get_random_profit()
    profit_wallets[0] = investor  # ensure known wallet gets share

    master = deploy_master(owner, profit_wallets, profit_perf, fee_manager.address)
    fee_manager.setMaster(master.address, {"from": owner})

    token = deploy_token(creator, master.address, weth.address, router.address)

    # Simulate a buy to trigger fee distribution
    amount_to_buy = 1000e18
    fee = get_buy_fee(token, amount_to_buy, master)
    tx = master.deploy(
        "Token", "T", b"metadata",
        weth.address, router.address, amount_to_buy,
        {"from": creator, "value": fee}
    )
    assert "TokenCreated" in tx.events
    token = tx.events["TokenCreated"][0]["token"]

    new_token = Contract.from_abi("Token", token, Token.abi)
    fee = get_buy_fee(new_token, amount_to_buy, master)
    new_token.buy(amount_to_buy, creator, {"from": creator, "value": fee})

    # Check fee was accounted properly
    creator_fee_balance = fee_manager.getCreatorBalance(creator)
    platform_fee_balance = fee_manager.getPlatformFeesAvailable()
    assert creator_fee_balance > 0
    assert platform_fee_balance > 0

    # Only creator can withdraw their fee
    with reverts("NoBalanceToWithdraw: "):
        fee_manager.withdrawCreatorFees({"from": someone})

    initial_balance = creator.balance()
```

```
    tx = fee_manager.withdrawCreatorFees({"from": creator})
    assert tx.events["CreatorFeeWithdrawn"]["creator"] == creator
    assert creator.balance() > initial_balance

    # Distribute platform fees to investors
    fee_manager.distributePlatformFees({"from": someone})
    investor_share = fee_manager.getInvestorBalance(investor)
    assert investor_share > 0

    # Withdraw investor share
    with reverts("NoBalanceToWithdraw: "):
        fee_manager.withdrawInvestorFees({"from": someone})

    initial_investor_balance = investor.balance()
    tx = fee_manager.withdrawInvestorFees({"from": investor})
    assert tx.events["InvestorFeeWithdrawn"]["investor"] == investor
    assert investor.balance() > initial_investor_balance


def test_set_master_and_failures(only_local):
    owner = get_account(0)
    attacker = get_account(1)

    fee_manager = deploy_fee_manager(owner)

    # Only owner can set master
    with reverts():
        fee_manager.setMaster(attacker, {"from": attacker})

    # Cannot set master to zero address
    with reverts("ZeroAddress: "):
        fee_manager.setMaster(ZERO_ADDRESS, {"from": owner})

    # Can only set master once
    fee_manager.setMaster(attacker, {"from": owner})
```

```
    with reverts("NotAuthorized: "):
        fee_manager.setMaster(attacker, {"from": owner})



def test_distribute_fee_validations(only_local):
    owner = get_account(0)
    creator = get_account(1)

    fee_manager = deploy_fee_manager(owner)
    profit_wallets, profit_perf = get_random_profit()
    master = deploy_master(owner, profit_wallets, profit_perf, fee_manager.address)
    fee_manager.setMaster(master.address, {"from": owner})

    token = deploy_token(creator, master.address, deploy_weth(owner).address,
deploy_router(owner, owner, ZERO_ADDRESS).address)

    # Cannot distribute fee without value
    with reverts("ZeroAmount: "):
        fee_manager.distributeFee(token.address, {"from": creator, "value": 0})

    # Cannot distribute fee if not locked
    new_fee_manager = deploy_fee_manager(owner)
    with reverts("NotAuthorized: "):
        new_fee_manager.distributeFee(token.address, {"from": creator, "value":
1e18})

    # Cannot distribute from zero token
    with reverts("InvalidToken: "):
        fee_manager.distributeFee(ZERO_ADDRESS, {"from": creator, "value": 1e18})
```

Token:

```
from brownie import (
    reverts,
    Token
)
```

```python
from brownie.network.contract import Contract


from scripts.helpful_scripts import (
    ZERO_ADDRESS,
    get_account,
    get_buy_fee,
    get_random_profit
)


from scripts.deploy import (
    deploy_weth,
    deploy_router,
    deploy_factory,
    deploy_fee_manager,
    deploy_token,
    deploy_master
)


def test_buy(only_local):
    # Arrange
    owner = get_account(0)
    other = get_account(1)
    extra = get_account(2)
    another = get_account(3)

    # Deploy contracts
    factory = deploy_factory(owner, owner)
    weth = deploy_weth(owner)
    router = deploy_router(owner, factory.address, weth.address)

    fee_manager = deploy_fee_manager(owner)
    profit_wallets, profit_perf = get_random_profit()
    master = deploy_master(owner, profit_wallets, profit_perf, fee_manager.address)
    fee_manager.setMaster(master.address, {"from": owner})
    token = deploy_token(owner, master.address, weth.address, router.address)
```

```
# simulate buy
result = token.simulateBuy(1000e18)
assert result[0] > 0  # value
assert result[1] > 0  # avgPrice
assert result[2] > 0  # endPrice

total = get_buy_fee(token, 1000e18, master)
tx = master.deploy(
    "Token", "T", b"metadata",
    weth.address, router.address, 1000e18,
    {"from": other, "value": total}
)
assert "TokenCreated" in tx.events
token = tx.events["TokenCreated"][0]["token"]

new_token = Contract.from_abi("Token", token, Token.abi)

with reverts("InsufficientFunds: "):
    new_token.buy(50e18, other, {"from": extra, "value": 0})

fee = get_buy_fee(new_token, 50e18, master)
tx = new_token.buy(50e18, other, {"from": extra, "value": fee})
assert tx.events['Transfer'][0]['from'] == ZERO_ADDRESS
assert tx.events['Transfer'][0]['to'] == other
assert tx.events['Transfer'][0]['value'] == 50e18

fee = get_buy_fee(new_token, 1000e18, master)
tx = new_token.buy(1000e18, extra, {"from": other, "value": fee})
assert tx.events['Transfer'][0]['from'] == ZERO_ADDRESS
assert tx.events['Transfer'][0]['to'] == extra
assert tx.events['Transfer'][0]['value'] == 1000e18

fee = get_buy_fee(new_token, 10e18, master)
tx = new_token.buy(10e18, another, {"from": another, "value": fee})
```

```
    assert tx.events['Transfer'][0]['from'] == ZERO_ADDRESS
    assert tx.events['Transfer'][0]['to'] == another
    assert tx.events['Transfer'][0]['value'] == 10e18

    with reverts("NotReady: "):
        new_token.list({"from": other})

    fee = get_buy_fee(new_token, 800000000e18 - 2060e18, master)
    tx = new_token.buy(800000000e18 - 2060e18, another, {"from": another, "value":
fee})
    assert tx.events['TokenListed'][0]['token'] == token

    with reverts("Untradable: "):
        new_token.buy(100e18, another, {"from": another, "value": fee})

    with reverts("NotReady: "):
        new_token.list({"from": other})

    with reverts("Untradable: "):
        new_token.sell(100e18, {"from": other})

def test_check_buy(only_local):
    # Arrange
    owner = get_account(0)
    other = get_account(1)
    extra = get_account(2)
    another = get_account(3)

    # Deploy contracts
    factory = deploy_factory(owner, owner)
    weth = deploy_weth(owner)
    router = deploy_router(owner, factory.address, weth.address)

    fee_manager = deploy_fee_manager(owner)
    profit_wallets, profit_perf = get_random_profit()
```

```
    master = deploy_master(owner, profit_wallets, profit_perf, fee_manager.address)
    fee_manager.setMaster(master.address, {"from": owner})
    token = deploy_token(owner, master.address, weth.address, router.address)

    total = get_buy_fee(token, 400e18, master)
    tx = master.deploy(
        "Token", "T", b"metadata",
        weth.address, router.address, 400e18,
        {"from": other, "value": total}
    )
    assert "TokenCreated" in tx.events
    token = tx.events["TokenCreated"][0]["token"]

    new_token = Contract.from_abi("Token", token, Token.abi)
    fee = get_buy_fee(new_token, 50e18, master)
    assert fee > 0
    tx = new_token.buy(50e18, extra, {"from": extra, "value": fee})
    assert tx.events['Transfer'][0]['from'] == ZERO_ADDRESS
    assert tx.events['Transfer'][0]['to'] == extra
    assert tx.events['Transfer'][0]['value'] == 50e18

def test_sell(only_local):
    # Arrange
    owner = get_account(0)
    other = get_account(1)
    extra = get_account(2)
    another = get_account(3)

    # Deploy contracts
    factory = deploy_factory(owner, owner)
    weth = deploy_weth(owner)
    router = deploy_router(owner, factory.address, weth.address)

    fee_manager = deploy_fee_manager(owner)
    profit_wallets, profit_perf = get_random_profit()
```

```
    master = deploy_master(owner, profit_wallets, profit_perf, fee_manager.address)
    fee_manager.setMaster(master.address, {"from": owner})
    token = deploy_token(owner, master.address, weth.address, router.address)

    total = get_buy_fee(token, 400e18, master)
    tx = master.deploy(
        "Token", "T", b"metadata",
        weth.address, router.address, 400e18,
        {"from": other, "value": total}
    )
    assert "TokenCreated" in tx.events
    token = tx.events["TokenCreated"][0]["token"]

    new_token = Contract.from_abi("Token", token, Token.abi)

    with reverts("InsufficientTokens: "):
        new_token.sell(10e18, {"from": extra})

    fee = get_buy_fee(new_token, 100e18, master)
    tx = new_token.buy(100e18, extra, {"from": extra, "value": fee})
    assert tx.events['Transfer'][0]['from'] == ZERO_ADDRESS
    assert tx.events['Transfer'][0]['to'] == extra
    assert tx.events['Transfer'][0]['value'] == 100e18

    balance_before = extra.balance()
    token_balance_before = new_token.balanceOf(extra)

    tx =  new_token.sell(10e18, {"from": extra})
    assert tx.events['Transfer'][0]['from'] == extra
    assert tx.events['Transfer'][0]['to'] == ZERO_ADDRESS
    assert tx.events['Transfer'][0]['value'] == 10e18
    balance_after = extra.balance()

    assert new_token.balanceOf(extra) == token_balance_before - 10e18
    assert balance_after > balance_before
```

Technical Findings Summary

Findings

Vulnerability Level	Total	Pending	Not Apply	Acknowledged	Partially Fixed	Fixed
High	0					
Medium	2					
Low	1					
Informational	0					

Assessment Results

Score Results

Review	Score
Global Score	**85/100**
Assure KYC	Not completed
Audit Score	85/100

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 PASS

Following our comprehensive security audit of the token contract for the OnlyPump project, the project did meet 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 OnlyPump 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 OnlyPump, 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 OnlyPumps 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 OnlyPump 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 OnlyPump may access, and depend upon, multiple layers of third parties.