SMART CONTRACT AUDIT

ZOKYO.

Jan 14th, 2022 | v. 1.0

PASS

Zokyo Security team has concluded that these smart contracts pass security qualifications and are fully production-ready

TECHNICAL SUMMARY

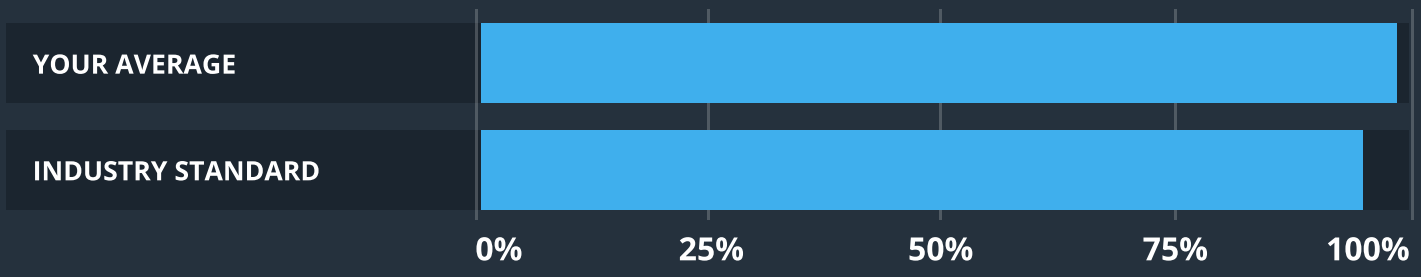
This document outlines the overall security of the 2PI smart contracts, evaluated by Zokyo's Blockchain Security team.

The scope of this audit was to analyze and document the 2PI smart contract codebase for quality, security, and correctness.

Contract Status

There was 1 critical issue found during the audit.

Testable Code



The testable code is 98.69%, which is above the industry standard of 95%.

It should be noted that this audit is not an endorsement of the reliability or effectiveness of the contract, rather limited to an assessment of the logic and implementation. In order to ensure a secure contract that's able to withstand the Ethereum network's fast-paced and rapidly changing environment, we at Zokyo recommend that the 2PI team put in place a bug bounty program to encourage further and active analysis of the smart contract.

TABLE OF CONTENTS

AUDITING STRATEGY AND TECHNIQUES APPLIED

The Smart contract's source code was taken from the 2PI repository.

Repository:
https://github.com/2pinetwork/contracts/tree/Zokyo-audit/

Last commit:
4be002bab5d3170ed4964a362519df11aae38868

Contracts under the scope:

- Archimedes;
- ArchimedesAPI;
- BridgedPiToken;
- Controller;
- ControllerAaveStrat;
- ControllerCurveStrat;
- Distributor;
- FeeManager;
- PiToken;
- PiVault;
- Referral;
- Swappable.

Throughout the review process, care was taken to ensure that the token contract:

- Implements and adheres to existing Token standards appropriately and effectively;
- Documentation and code comments match logic and behavior;
- Distributes tokens in a manner that matches calculations;
- Follows best practices in efficient use of gas, without unnecessary waste;
- Uses methods safe from reentrance attacks;
- Is not affected by the latest vulnerabilities;
- Whether the code meets best practices in code readability, etc.

Zokyo's Security Team has followed best practices and industry-standard techniques to verify the implementation of smart contracts. To do so, the code is reviewed line-by-line by our smart contract developers, documenting any issues as they are discovered. Part of this work includes writing a unit test suite using the Truffle testing framework. In summary, our strategies consist largely of manual collaboration between multiple team members at each stage of the review:

1	Due diligence in assessing the overall code quality of the codebase.
2	Cross-comparison with other, similar smart contracts by industry leaders.
3	Testing contract logic against common and uncommon attack vectors.
4	Thorough, manual review of the codebase, line-by-line.

SUMMARY

The Zokyo team has conducted a security audit of the given codebase. The contracts provided for an audit are well written and structured. All the findings within the auditing process are presented in this document.

Worth mentioning that we didn't find any critical issue. During the auditing process, our auditor's team found 1 issue with a high severity level, 1 issue with a low severity level, and 14 informational issues. All issues were resolved by the 2PI team.

Based on the results of the audit, we can give a score of 98 and state that the audited contracts are fully production-ready.

STRUCTURE AND ORGANIZATION OF DOCUMENT

For ease of navigation, sections are arranged from most critical to least critical. Issues are tagged “Resolved” or “Unresolved” depending on whether they have been fixed or addressed. Furthermore, the severity of each issue is written as assessed by the risk of exploitation or other unexpected or otherwise unsafe behavior:

Critical

The issue affects the ability of the contract to compile or operate in a significant way.

High

The issue affects the ability of the contract to compile or operate in a significant way.

Medium

The issue affects the ability of the contract to operate in a way that doesn’t significantly hinder its behavior.

Low

The issue has minimal impact on the contract’s ability to operate.

Informational

The issue has no impact on the contract’s ability to operate.

COMPLETE ANALYSIS

HIGH | RESOLVED

Function deposit on ControllerLPWithoutStrat have an empty body.

Recommendation:
Implement the body or delete the function.

LOW | RESOLVED

On the Controller contract, there is a line to check if the contract is an ERC20, just by checking the symbol function is a very naive way of doing it.

Recommendation:
Implement the EIP-165 to check if a contract is compliant with a standard, in this case, the standard is ERC20.

INFORMATIONAL | RESOLVED

The verification of controller address from Archimedes line 318 and line 331 is not in line with best practices.

Recommendation:
Move the verification of controller address in a modifier to be compliant with best practices.

INFORMATIONAL | RESOLVED

Not all revert messages from the required functions are optimized.

Recommendation:
Be sure all the revert messages from the required functions are 32bytes long for gas optimization.

INFORMATIONAL | RESOLVED

On the Distributor contract, the functions depositToFounders, depositToInvestors implement a loop without a safety check condition.

Recommendation:
Add a condition in the loop to stop the looping if the gas left is below 20 000 units.

INFORMATIONAL | RESOLVED

On the ControllerAaveStrat contract, the functions _leverage, _fullDelevrage implement a loop without a safety check condition.

Recommendation:
Add a condition in the loop to stop the looping if the gas left is below 20 000 units.

INFORMATIONAL | RESOLVED

On the Archimedes contract, the functions massUpdatePools, harvestAll implement a for loop without a safety check condition.

Recommendation:
Add a condition in the loop to stop the looping if the gas left is below 20 000 units.

INFORMATIONAL | RESOLVED

Not all the functions names respect the same pattern, it's part of bad practices to have different naming conventions for the same function types.

Recommendation:
To be fully compliant with best practices ensure that all internal functions name starts with '_' and all public or external function start with normal letters.

INFORMATIONAL | RESOLVED

On the ControllerCurveStrat contract, the setPerformanceFee, setPoolMInVirtualPrice, setPoolSlippagePrice, setRatioForFullWIthdraw functionalities need more sanity checks.

Recommendation:
Even if the function can only be called by the admin, it does not mean the admin can not make a mistake, add some sanity checks using require, for example, check if the new value is different from the old one to not use gas to set the same value twice.

INFORMATIONAL | RESOLVED

On the Controller contract, the setDepositCap, setWIthdrawFee, setStrategy, setTreasury, setFarmPid functionalities need more sanity checks.

Recommendation:
Even if the function can only be called by the admin, it does not mean the admin can not make a mistake, add some sanity checks using require, for example, check if the new value is different from the old one to not use gas to set the same value twice.

INFORMATIONAL | RESOLVED

On the Swappable contract, the setSwapSlippaeRation and setMaxPriceOffset functionalities need more sanity checks.

Recommendation:
Even if the function can only be called by the admin, it does not mean the admin can not make a mistake, add some sanity checks using require, for example, check if the new value is different from the old one to not use gas to set the same value twice.

INFORMATIONAL | RESOLVED

On the FeeManager contract, the setTreasuryRation, setTreasury, setExchange functionalities need more sanity checks.

Recommendation:
Even if the function can only be called by the admin, it does not mean the admin can not make a mistake, add some sanity checks using require, for example, check if the new value is different from the old one to not use gas to set the same value twice.

INFORMATIONAL | RESOLVED

On the Controller contract, the deposit functionality needs more sanity checks.

Recommendation:
Should check with a required function if the amount is bigger than 0, so it will revert and not use gas useless for users.

INFORMATIONAL | RESOLVED

To follow best practices recommendation, move the ICurvePool and IRewardsGauce in separate files.

Recommendation:
Create new files for each interface and import the new interfaces in the contracts that will use them.

INFORMATIONAL | RESOLVED

To follow best practices recommendation, move the IStrategy and IFarm in separate files.

Recommendation:
Create new files for each interface and import the new interfaces in the contracts that will use them.

INFORMATIONAL | RESOLVED

Specifying a pragma version with the caret symbol (^) upfront which tells the compiler to use any version of solidity bigger than specified is considered not a good practice.

Recommendation:
Change the caret symbol (^) to an equal symbol (=), to ensure that contracts will be compiled only with the specified compiler version.

	Archimedes; **ArchimedesAPI;** **BridgedPiToken;** **Controller;** **ControllerAaveStrat;** **ControllerCurveStrat;**	**Distributor;** **FeeManager;** **PiToken;** **PiVault;** **Referral;** **Swappable.**
Re-entrancy	Pass	Pass
Access Management Hierarchy	Pass	Pass
Arithmetic Over/Under Flows	Pass	Pass
Unexpected Ether	Pass	Pass
Delegatecall	Pass	Pass
Default Public Visibility	Pass	Pass
Hidden Malicious Code	Pass	Pass
Entropy Illusion (Lack of Randomness)	Pass	Pass
External Contract Referencing	Pass	Pass
Short Address/ Parameter Attack	Pass	Pass
Unchecked CALL Return Values	Pass	Pass
Race Conditions / Front Running	Pass	Pass
General Denial Of Service (DOS)	Pass	Pass
Uninitialized Storage Pointers	Pass	Pass
Floating Points and Precision	Pass	Pass
Tx.Origin Authentication	Pass	Pass
Signatures Replay	Pass	Pass
Pool Asset Security (backdoors in the underlying ERC-20)	Pass	Pass

CODE COVERAGE AND TEST RESULTS FOR ALL FILES

Tests written by 2PI team

Code Coverage

The resulting code coverage (i.e., the ratio of tests-to-code) is as follows:

FILE	% STMTS	% BRANCH	% FUNCS	% LINES	UNCOVERED LINES
contracts/	96.29	81.51	97.93	98.14	
Archimedes.sol	96.53	84.09	97.56	97.60	176, 177, 425, 449
ArchimedesAPI.sol	97.59	84.69	97.30	97.50	185, 186, 306, 396
BridgePiToken.sol	98.48	92.86	100.00	100.00	
Controller.sol	98.78	78.33	100.00	100.00	158, 161, 406, 409
ControllerAaveStrat.sol	93.60	75.00	97.06	97.45	... 140, 279, 292
ControllerCurveStrat.sol	93.71	76.67	94.44	94.78	
Distributor.sol	98.68	81.82	100.00	100.00	
FeeManager.sol	100.00	73.33	100.00	100.00	
PiAdmin.sol	100.00	100.00	100.00	100.00	
PiToken.sol	100.00	94.23	100.00	100.00	
PiVault.sol	100.00	92.31	100.00	100.00	
Refferal.sol	100.00	50.00	100.00	100.00	
Swappable.sol	100.00	57.14	100.00	100.00	
All files	**96.92**	**81.51**	**97.93**	**98.14**	

Test Results

Archimedes setup

- ✓ should revert of 0 address piToken
- ✓ should revert for old block number

Archimedes

addNewPool

- ✓ Should reverse with zero address want
- ✓ Should reverse with non-archimedes controller (4216ms)
- ✓ Should reverse for controller without strategy

changePoolWeighing

- ✓ Should update totalWeighing

pendingPiToken

- ✓ should return 0 for future block
- ✓ should return 0 for unknown user

deposit

- ✓ should work with LP (5646ms)

FullFlow

- ✓ Full flow with 2 accounts && just 1 referral (19472ms)

depositNative

- ✓ should revert for depositAll
- ✓ Should revert with 0 amount
- ✓ Should revert for not wnative pool
- ✓ Should get wnative shares and then withdraw (25878ms)

withdraw

- ✓ Should revert with 0 shares
- ✓ Should revert without shares

getPricePerFullShare

- ✓ Should get 1e18 for 0 shares
- ✓ Should get updated value after deposit (4293ms)

decimals

- ✓ Should be controller decimals

balance & balanceOf

- ✓ Should get 0 for 0 shares (1845ms)
- ✓ Should get 1 for 1 shares (6876ms)

deposit

- ✓ should revert with 0 amount

```
    ✓ should deposit all balance (5466ms)
  setReferralCommissionRate
    ✓ should revert from not admin change
    ✓ should change rate from 1 to 2
    ✓ should revert for maximum referral rate
  Token decimals
    ✓ Should have the same decimals than want (16846ms)
  Harvest
    ✓ should not receive double reward harvest (8853ms)

ArchimedesAPI setup
  ✓ should revert for 0 address piToken
  ✓ should revert for old block number
  ✓ should revert for 0 address handler

ArchimedesAPI
  setExchange
    ✓ should be reverted for non admin
    ✓ should be reverted for 0 address
  setHandler
    ✓ should revert for 0 address
    ✓ should revert for non admin
    ✓ should change handler
  addNewPool
    ✓ Should reverse with zero address want
    ✓ Should reverse with non-archimedes controller (3109ms)
    ✓ Should reverse for controller without strategy
  changePoolWeighing
    ✓ Should update totalWeighing
  FullFlow
    ✓ with 2 accounts && just 1 referral (39212ms)
  withdraw
    ✓ Should revert with 0 shares
    ✓ Should revert without shares
  getPricePerFullShare
    ✓ Should get 1e18 for 0 shares
    ✓ Should get updated value after deposit (4637ms)
  decimals
    ✓ Should be controller decimals
```

balance & balanceOf
- ✓ Should get 0 for 0 shares (2310ms)
- ✓ Should get 1 for 1 shares (4054ms)

deposit
- ✓ should revert with 0 amount

setReferralCommissionRate
- ✓ should revert from not admin change
- ✓ should change rate from 1 to 2
- ✓ should revert for maximum referral rate

Token decimals
- ✓ Should have the same decimals than want (20363ms)

setRoute
- ✓ should be reverted for non piToken first token
- ✓ should be reverted for non want last token

emergencyWithdraw
- ✓ should be reverted for not auth user

Controller Aave Strat

PerformanceFee
- ✓ should be collected (23615ms)

Controller Curve Strat
- ✓ Full deposit + harvest strat + withdraw (72800ms)

PiOracle
- ✓ should get less price each swap
- ✓ should get more price each swap

UniZap
- ✓ should get USDC-USDT with BTC (23781ms)
- ✓ should get BTC-USDC with BTC (6477ms)
- ✓ should get USDC with BTC

63 passing (10m)

Tests written by Zokyo Security team

As part of our work assisting 2PI in verifying the correctness of their contract code, our team was responsible for writing integration tests using the Truffle testing framework.

Tests were based on the functionality of the code, as well as a review of the 2PI contract requirements for details about issuance amounts and how the system handles these.

Code Coverage

The resulting code coverage (i.e., the ratio of tests-to-code) is as follows:

FILE	% STMTS	% BRANCH	% FUNCS	% LINES	UNCOVERED LINES
contracts/	98.69	94.68	98.74	99.02	
Archimedes.sol	96.53	92.05	97.56	97.02	... 383, 385, 483
ArchimedesAPI.sol	98.80	93.88	100.00	99.38	323
BridgePiToken.sol	100.00	100.00	100.00	100.00	
Controller.sol	98.78	93.33	100.00	98.75	151
ControllerAaveStrat.sol	99.42	90.22	100.00	100.00	
ControllerCurveStrat.sol	99.30	95.56	100.00	100.00	
Distributor.sol	100.00	100.00	100.00	100.00	
FeeManager.sol	100.00	100.00	100.00	100.00	
PiToken.sol	96.30	94.23	90.48	96.15	250, 252, 265
PiVault.sol	100.00	100.00	100.00	100.00	
Refferal.sol	100.00	100.00	100.00	100.00	
Swappable.sol	100.00	100.00	100.00	100.00	
All files	**98.69**	**94.68**	**98.74**	**99.02**	

Test Results

Contract: Archimedes

- ✓ should fail contract deploy
- ✓ should add new pool, fail, address zero not allowed
- ✓ should add new pool, fail, not an archimedes controller
- ✓ should add new pool, fail, not an strategy
- ✓ should fail, mass update false
- ✓ should fail, mass update false
- ✓ should work, mass update false
- ✓ should change pool weight, mass update false, pi token community left to mint zero (919ms)
- ✓ should change pool weight, mass update true, pi token community left to mint zero (1721ms)
- ✓ should call pending pi token (1625ms)
- ✓ should call deposit native, should fail, insufficient deposit (1300ms)
- ✓ should call deposit native, should fail, only native token pool (1299ms)
- ✓ should call deposit native, should work (1979ms)
- ✓ should call deposit, fail, insufficient deposit (2617ms)
- ✓ should call deposit, fail (2330ms)
- ✓ should call deposit all, fail, can't deposit all native (2680ms)
- ✓ should call deposit all, should work (2649ms)
- ✓ should call withdraw, fail, 0 shares (2282ms)
- ✓ should call withdraw, fail, no sufficient shares (2972ms)
- ✓ should call withdraw, fail, no funds withdraw (2400ms)
- ✓ should call withdraw, funds to withdraw, pool address not wnative (2697ms)
- ✓ should call withdraw, funds to withdraw, pool address is wnative (2386ms)
- ✓ should call withdraw all, should work (2060ms)
- ✓ should call harvest, should work, not emitting harvest event (2372ms)
- ✓ should call set referral address, fail, same manager (1283ms)
- ✓ should call set referral address, fail address is zero (944ms)
- ✓ should call set referral address, work (1608ms)
- ✓ should call set referral commision, fail same rate (904ms)
- ✓ should call set referral commision, fail bigger then maximum rate (1270ms)
- ✓ should call poolLength, should work
- ✓ should call user shares without address, work (1653ms)
- ✓ should call user shares with address, work (3241ms)
- ✓ call paid rewards after harvesting (2455ms)
- ✓ call price per share (2729ms)
- ✓ call balance on controller (1709ms)

- ✓ should call balance of on controller (2029ms)
- ✓ should call block number on archimedes
- ✓ should call pi token per block
- ✓ should call redeem pi token, fail still minting (919ms)
- ✓ should call redeem pi token, fail still waiting
- ✓ should call redeem pi token, should work, balance 0
- ✓ should call redeem pi token, should work, balance greater then 0
- ✓ should call harvest all, should work, not emitting harvest event (2055ms)
- ✓ should call emergency withdraw, should work (2057ms)
- ✓ should call before shares transfer, amount 0 return empty (1788ms)
- ✓ should call before shares transfer, amount bigger then 0 should work (2350ms)
- ✓ should call after shares transfer, amount bigger then 0 should work (1910ms)
- ✓ should call after shares transfer, amount 0 should return empty (1679ms)
- ✓ should call safe pi token transfer, amount bigger then pi token balance (935ms)
- ✓ should call safe pi token transfer, amount bigger then pi token balance
- ✓ should call set referral commision, work (937ms)
- ✓ should call get multiplier
- ✓ should call pay referral commision, exit at first if (2470ms)
- ✓ should call pay referral commision, go into first if (2228ms)
- ✓ should call pay referral commision, go into last if (2666ms)
- ✓ should call pay referral commision, refer address is zero (2446ms)
- ✓ should call pending pi token (952ms)
- ✓ should call update pool, community left to mint bigger then 0 (978ms)
- ✓ should call update pool, shares total 0 (1324ms)
- ✓ should call update pool, pi token reward 0 (1032ms)
- ✓ should call before shares transfer, not an controller (1737ms)

Contract: ArchimedesAPI

- ✓ should deploy, all fail
- ✓ should deploy, works
- ✓ should set exchange, fail same address
- ✓ should set exchange, fail zero address
- ✓ should set exchange, works
- ✓ should set handler, fail same address
- ✓ should set handler, fail zero address
- ✓ should set handler, works
- ✓ should set route, fail first token not pi
- ✓ should call route, fail last token not want (973ms)

- ✓ should set route, fail unknown pool (1312ms)
- ✓ should set route, works (971ms)
- ✓ should call add new pool, all fails (967ms)
- ✓ should call add new pool, mass update false, pid doesn't match (988ms)
- ✓ should call add new pool, mass update false, pid match (1307ms)
- ✓ should call add new pool, mass update true, pid match (989ms)
- ✓ should call change pool weight, mass update false (1261ms)
- ✓ should call change pool weight, mass update true (1295ms)
- ✓ should call update pool, api left ot mint bigger then 0, controller total supply 0 (1286ms)
- ✓ should call update pool, block number equal pool last reward block (1641ms)
- ✓ should call update pool, api left ot mint bigger then 0, controller total supply bigger then 0 (1281ms)
- ✓ should call update pool, api left ot mint bigger then 0, controller total supply bigger then 0, pi tokens rewards bigger then 0 (1308ms)
- ✓ should call update pool, api left ot mint bigger then 0, controller total supply bigger then 0, pi tokens rewards bigger then 0, rewards bigger then left to mint (1278ms)
- ✓ should call deposit, fail insufficient amount (1039ms)
- ✓ should call deposit, fail not handler (1032ms)
- ✓ should call deposit, user shares 0 (1972ms)
- ✓ should call deposit, user shares bigger then 0, pending bigger then 0 (1649ms)
- ✓ should call deposit, user shares bigger then 0, pending 0 (1614ms)
- ✓ should call deposit, user shares bigger then 0, pending bigger then 0, go into swap for want (2569ms)
- ✓ should call deposit, user shares bigger then 0, pending 0 (1650ms)
- ✓ should call withdraw, fail 0 shares (1051ms)
- ✓ should call withdraw, user shares 1, fail not sufficient funds (1387ms)
- ✓ should call withdraw, user shares 1, fail can't withdraw from controller (1677ms)
- ✓ should call withdraw, user shares 1, worked (2740ms)
- ✓ should call emergency withdraw, all user shares, worked (1753ms)
- ✓ should call emergency withdraw, all user shares, fail not authorized (1714ms)
- ✓ should call harvest all, worked, emit harvested (2716ms)
- ✓ should call harvest all, worked, shares 0, get out at first (1729ms)
- ✓ should call harvest all, worked, balance bigger then 0, go in deposit controller, harvested emitted (2768ms)
- ✓ should call before shares transfer and after shares transfer reverts (1025ms)
- ✓ should set referral, fail same address (1044ms)
- ✓ should set referral, fail zero address (1030ms)
- ✓ should set referral, works (1056ms)
- ✓ should set referral, fail same address (1023ms)

- ✓ should set referral, fail zero address
- ✓ should set referral, works (1037ms)
- ✓ call the views, check (1782ms)
- ✓ call redeem stuck pi tokens, fail still minting (1647ms)
- ✓ call redeem stuck pi tokens, fail still waiting (2329ms)
- ✓ call redeem stuck pi tokens, balance 0, not go into if (1327ms)
- ✓ call redeem stuck pi tokens, balance greater than 0, go into if (2021ms)
- ✓ call pay refferal commision, referral mgr valid, refferal commision rate bigger than 0, commision 0 (993ms)
- ✓ call pay refferal commision, referral mgr valid, refferal commision rate bigger than 0, commision bigger than 0, api smaller than commision (1945ms)
- ✓ call pay refferal commision, referral mgr valid, refferal commision rate bigger than 0, commision bigger than 0, api bigger than commision (2957ms)
- ✓ check block number, work

Contract: BridgePiToken

- ✓ deploy new bridge, should work (923ms)
- ✓ deploy new bridge, should fail, already set
- ✓ should set community mint per block, fail same rate (940ms)
- ✓ should set community mint per block, work (935ms)
- ✓ should api rate mint per block, fail same rate (950ms)
- ✓ should api rate mint per block, work (970ms)
- ✓ should add minter, work
- ✓ should call available, work
- ✓ should call community mint, fail only minters
- ✓ should call community mint, can not mint to zero address
- ✓ should call community mint, fail insufficient supply (992ms)
- ✓ should call community mint, rewards not initialized (992ms)
- ✓ should call community mint, still waiting for rewards block
- ✓ should call community mint, can't mint more than available (968ms)
- ✓ should call community mint, fail mint ration is zero
- ✓ should call community mint, work
- ✓ should call api mint, work (1045ms)
- ✓ should call community left to mint
- ✓ should call api left to mint (1041ms)
- ✓ should call balance of, work (954ms)
- ✓ should call api mint, fail can't mint more then expected
- ✓ should call community mint, fail can't mint more then expected (1030ms)

✓ should call api mint, work, old tranche (1086ms)
✓ should call community mint, work, old tranche
✓ should call before change min rate, into if
✓ should call before change min rate, into if (998ms)
✓ should call left to mint for current block, into if
✓ should call left to mint, into if
✓ check block number, work

Contract: Controller
✓ deploy fail, invalid erc20 from balanceOf
✓ deploy fail, invalid erc20 from allowance
✓ deploy fail, invalid pitoken on archimedes (1444ms)
✓ deploy fail, invalid treasury (1432ms)
✓ deploy works (1172ms)
✓ call decimals (1797ms)
✓ call set pid from archimedes, work (1785ms)
✓ call set pid from archimedes, fail (1824ms)
✓ call set pid from archimedes, fail, not as archimedes (2444ms)
✓ call set treasury, fail same address (1447ms)
✓ call set treasury, fail zero address (1471ms)
✓ call set treasury, work (1833ms)
✓ call set strategy, fail same strategy (1821ms)
✓ call set strategy, fail (1499ms)
✓ call set strategy, fail old strategy still has deposits (2294ms)
✓ call set strategy, work (2559ms)
✓ call set withdraw fee, fail same rate (1096ms)
✓ call set withdraw fee, fail bigger than maximum (1770ms)
✓ call set withdraw fee, work (2159ms)
✓ call set deposit cap, fail same cap (1466ms)
✓ call set deposit cap, work (1097ms)
✓ call deposit, from archimedes, fail strategy paused (2395ms)
✓ call deposit, from archimedes, fail insufficient amount (2220ms)
✓ call deposit, from archimedes, works (2590ms)
✓ call withdraw, from archimedes, fail insufficient shares (2874ms)
✓ call withdraw, from archimedes, fail can't withdraw from strategy (2276ms)
✓ call withdraw, from archimedes (2342ms)
✓ call check deposit cap, from archimedes (1870ms)
✓ call check deposit cap, work (1845ms)

- ✓ call check available deposit, from archimedes, deposit cap bigger than 0, balance bigger then deposit (2158ms)
- ✓ call check available deposit, from archimedes, deposit cap 0 (1490ms)
- ✓ call check available deposit, from archimedes, deposit cap bigger than 0, balance lower than deposit (2140ms)
- ✓ call before token transfer, go into if, from archimedes (1210ms)
- ✓ call after token transfer, go into if, from archimedes (1239ms)
- ✓ call strategy deposit with amount bigger than 0 (3228ms)
- ✓ call set pid, fail already assigned (1855ms)
- ✓ set strategy works, not go into if (2129ms)

Contract: ControllerAaveStrat

- ✓ deploy want fail address zero
- ✓ deploy controller fail address zero
- ✓ deploy treasury fail address zero
- ✓ deploy borrow rate bigger then borrow rate max fail
- ✓ deploy borrow rate max bigger then ratio precision fail
- ✓ deploy works
- ✓ call set treasury, fail same address
- ✓ call set treasury, fail zero address
- ✓ call set treasury, works
- ✓ call set exchange, fail same address
- ✓ call set exchange, fail zero address
- ✓ call set exchange, works
- ✓ call set swap route, fail route[0] is not wnative
- ✓ call set swap route, fail last route is not want (782ms)
- ✓ call set swap route, works
- ✓ call performance fee, fail same address
- ✓ call set ratio for full withdraw, same ratio
- ✓ call set ratio for full withdraw, fail more than 100%
- ✓ call set ratio for full withdraw, works
- ✓ call performance fee, fail more than 100%
- ✓ call performance fee, works
- ✓ should call before movement, ... (2083ms)
- ✓ should call before movement, fee bigger than 0 (2341ms)
- ✓ should call before movement, current balance smaller than last balance (1770ms)
- ✓ should call before movement, perfFee bigger than balance (3125ms)
- ✓ should call deposit, minLeverage smaller than amount (1615ms)

- ✓ should call deposit, minLeverage bigger than amount (2233ms)
- ✓ should call deposit, minLeverage smaller than amount, borrowDepth 1000 (28353ms)
- ✓ should call withdraw, balance bigger than amount (2281ms)
- ✓ should call withdraw, balance lower than amount, borrow balance equal 0 (2014ms)
- ✓ should call withdraw, full deleverage path (2041ms)
- ✓ should call withdraw, partial deleverage path, not go into while, toWithdraw smaller than wantBalance (2496ms)
- ✓ should call withdraw, partial deleverage path, go into while, go into withdraw and repay (1915ms)
- ✓ should call increase health factor, by ration bigger then 100% (790ms)
- ✓ should call increase health factor, borrow balance smaller than 0 (1167ms)
- ✓ should call increase health factor, borrow balance bigger then 0, toWithdraw 0 (1475ms)
- ✓ should call increase health factor, borrow balance bigger then 0, toWithdraw bigger than 0 (1126ms)
- ✓ should call rebalance, fail exceeds max borrow rate (797ms)
- ✓ should call rebalance, fail exceeds max borrow depth (1118ms)
- ✓ should call rebalance, works not go into last if (1959ms)
- ✓ should call rebalance, works not go last if (1277ms)
- ✓ should call panic, works (1285ms)
- ✓ should call retire strategy, works (3411ms)
- ✓ should call swap rewards internal function (2430ms)
- ✓ call charge fees, not go into if (781ms)
- ✓ call charge fees, go into if (802ms)
- ✓ call check unpause (1690ms)
- ✓ should call retire strategy, works, with paused, balance of pool 0 (2537ms)
- ✓ should call retire strategy, fail, only controller (1521ms)
- ✓ should call retire strategy, works, want differ from wnative (1855ms)

Contract: ControllerCurveStrat

- ✓ should deploy fail with zero addresses
- ✓ should deploy works
- ✓ should call set treasury fail same address
- ✓ should call set treasury fail zero address (1016ms)
- ✓ should call set treasury, work
- ✓ should call set exchange fail same address
- ✓ should call set exchange fail zero address (1031ms)
- ✓ should call set exchange, work
- ✓ should call set wnative swap route fail same address
- ✓ should call set wnative swap route fail zero address

- ✓ should call set wnative swap route, work (1100ms)
- ✓ should call set crv swap route fail same address (1050ms)
- ✓ should call set crv swap route fail zero address (1082ms)
- ✓ should call set crv swap route, work (782ms)
- ✓ should call set performance fee fail same address
- ✓ should call set performance fee fail zero address
- ✓ should call set performance feex, work
- ✓ should call set setPoolMinVirtualPrice fail same ratio (1057ms)
- ✓ should call set setPoolMinVirtualPrice fail can not be more than 100%
- ✓ should call set setPoolMinVirtualPrice feex, work (1066ms)
- ✓ should call set setPoolSlippageRatio fail same ratio (1077ms)
- ✓ should call set setPoolSlippageRatio fail can not be more than 100%
- ✓ should call set setPoolSlippageRatio feex, work (807ms)
- ✓ should call set setRatioForFullWithdraw fail same ratio
- ✓ should call set setRatioForFullWithdraw fail can not be more than 100%
- ✓ should call set setRatioForFullWithdraw feex, work
- ✓ should call before movement, current balance lower than last balance (1345ms)
- ✓ should call before movement, current balance higher than last balance, perf fee 0 (1087ms)
- ✓ should call before movement, current balance higher than last balance, perf fee bigger then 0, go into last if (1803ms)
- ✓ should call before movement, current balance higher than last balance, perf fee bigger then 0, go into second if (1781ms)
- ✓ should call before movement, current balance higher than last balance, perf fee bigger then 0, go into first if, fail remove liq exp 0 (2490ms)
- ✓ should call before movement, current balance higher than last balance, perf fee bigger then 0, go into first if, don't trigge require (1532ms)
- ✓ should call deposit, btc and btccrv balance both 0 (1789ms)
- ✓ should call deposit, btc balance bigger 0 (1572ms)
- ✓ should call deposit, btccrv balance bigger than 0 (1505ms)
- ✓ should call withdraw, balance bigger than amount (1215ms)
- ✓ should call withdraw, balance lower than amount, go in first if (1957ms)
- ✓ should call withdraw, balance lower than amount, go in first else, require liquidity hit (1513ms)
- ✓ should call withdraw, balance smaller then amount, dodge last if (1305ms)
- ✓ should call harvest, work simple (1781ms)
- ✓ should call harvest, wnative greater than 0, crv greater than 0 (2598ms)
- ✓ should call retire strat to harvest, with paused, wnative greater than 0, crv greater than 0 (3194ms)
- ✓ should call retire strat to harvest, without paused, wnative greater than 0, crv greater than 0 (3351ms)

- ✓ should call panic, wnative greater than 0, crv greater than 0 (1838ms)
- ✓ should call charge fees internal function, go into if (1078ms)
- ✓ should call charge fees internal function, not go into if (1060ms)
- ✓ should call retire strat to harvest, without paused, wnative greater than 0, crv greater than 0, expected equal 0 (2355ms)
- ✓ should call retire strat to harvest, not from controller, fail

Contract: Distributor

- ✓ should deploy
- ✓ should set treasury, fail same address
- ✓ should set treasury, fail zero address
- ✓ should set treasury, works (1038ms)
- ✓ should call distribute, fail have to wait (754ms)
- ✓ should call distribute, fail nothing more to do
- ✓ should call distribute, fail nothing more to do, multiplier 0, tokens for investors 0 (1785ms)
- ✓ should call distribute, fail nothing more to do, multiplier 0, tokens for investors bigger then 0 (1629ms)
- ✓ should call distribute, fail nothing more to do, multiplier 0, tokens for founder 0 (1889ms)
- ✓ should call distribute, fail nothing more to do, multiplier 0, tokens for treasury 0 (1645ms)
- ✓ should call distribute, fail nothing more to do, multiplier 10, tokens for treasury and founders 1 (1968ms)
- ✓ should deployd call distribute, fail nothing more to do, multiplier 10, tokens for investors 1 (2007ms)

Contract: FeeManager

- ✓ deploy fail treasury zero address (954ms)
- ✓ deploy success (1095ms)
- ✓ should call harvest, balance 0 (2036ms)
- ✓ should call harvest, balance greater than 0, not native (3107ms)
- ✓ set treasury, fail same address (1360ms)
- ✓ set treasury, fail zero address (1362ms)
- ✓ set treasury, works (1674ms)
- ✓ set treasury ratio, fail same ratio (1712ms)
- ✓ set treasury, fail greater than 50% (1015ms)
- ✓ set treasury, works (1404ms)
- ✓ set exchange, fail same address (1769ms)
- ✓ set exchange, fail zero address (1738ms)
- ✓ set exchange, works (1702ms)
- ✓ set route, fail address zero (1377ms)

- ✓ set route, fail invalid router, router smaller than 2 (1356ms)
- ✓ set route, fail invalid router, router smaller than 2 (1758ms)
- ✓ set route, fail invalid router, works (1388ms)
- ✓ should call harvest, balance greater than 0, route length bigger than 0 (2541ms)
- ✓ should call harvest, balance greater than 0, native true (2908ms)

Contract: PiToken

- ✓ check initialize (3329ms)
- ✓ check add mitner (2382ms)
- ✓ call init rewards, fail already set (2033ms)
- ✓ call init rewards, worked (2044ms)
- ✓ should set comm mint per block, fail same rate (1724ms)
- ✓ should set comm mint per block worked, go into if (2042ms)
- ✓ should set comm mint per block worked, not go into if (1382ms)
- ✓ should set api mint per block, fail same rate (1725ms)
- ✓ should set api mint per block worked, not go into if (1693ms)
- ✓ should mint for multichain, fail, insufficient supply (1372ms)
- ✓ should mint for multichain, fail, insufficient supply (1389ms)
- ✓ should mint for multichain, fail (1455ms)
- ✓ should call community mint, fail, only minters (1365ms)
- ✓ should call community mint, fail, can not mint to zero address (1740ms)
- ✓ should call community mint, fail, insufficient supply (1396ms)
- ✓ should call community mint, fail, rewards not initialized (1422ms)
- ✓ should call community mint, fail, still waiting for rewards block (1735ms)
- ✓ should call community mint, fail, mint capped (1436ms)
- ✓ should call community mint, fail, mint ratio is 0 (1760ms)
- ✓ should call community mint, works, to mint smaller than max mintable (1819ms)
- ✓ should call community mint, fail, to mint bigger then max mintable, can't mint more than expected (2437ms)
- ✓ should call community mint, works, to mint bigger then max mintable (1455ms)
- ✓ should call api mint, fail, to mint bigger then max mintable, can't mint more than expected (2109ms)
- ✓ should call api mint, works, to mint bigger than max mintable (2161ms)
- ✓ should call api mint, works, to mint smaller than max mintable (2167ms)
- ✓ should call community left to mint, works, to mint bigger than max mintable (2096ms)
- ✓ should call community left to mint, total left 0 (1796ms)
- ✓ should call community left to mint, tranches block 0 (2106ms)
- ✓ should call api left to mint, works, to mint bigger than max mintable (1763ms)

✓ should call tokens received, fail invalid token (1715ms)
✓ should check add burne and burn (1718ms)
✓ should check cap (2402ms)
✓ check block number, work (2053ms)

Contract: PiVault
✓ should add investor, check (1047ms)
✓ should add founder, check (1059ms)
✓ should call deposit all (1419ms)
✓ should call deposit all (1428ms)
✓ should call withdraw all, works, go into check withdraw but don't trigger any ifs (1387ms)
✓ should call withdraw all, investor still locked (1691ms)
✓ should call withdraw all, investor not locked (1398ms)
✓ should call withdraw all, founder still locked (1363ms)
✓ should call withdraw all, founder not locked, max withdraw not reached (1042ms)
✓ should call withdraw, amount not available (1056ms)
✓ should call withdraw, hit max withdraw (1366ms)
✓ should call withdraw, not hit max withdraw (1056ms)
✓ should call get price per full share (944ms)
✓ should call before token transfer, go into if, investors true, still locked
✓ should call before token transfer, go into if, investors true, not locked
✓ should call before token transfer, go into if, investors false, founder true, still locked
✓ should call before token transfer, go into if, investors false, founder true, not locked (1065ms)
✓ should call before token transfer, go into if, investors false, founder false (1030ms)

Contract: Referral
✓ deploy fail archimedes zero address
✓ should call recordReferral, not go into if (766ms)
✓ should call recordReferral, go into if (1059ms)
✓ should call recordReferral, fail not archimedes (1022ms)
✓ should call referral paid (1436ms)
✓ should check get referral (1100ms)

Contract: Swappable
✓ should set swap slippage ratio, fail same ratio (1083ms)
✓ should set swap slippage ratio, fail more than 100%
✓ should set swap slippage ratio should work
✓ should set max price offset, fail same ratio
✓ should set max price offset, fail more than 100%

```
✓ should set max price offset should work
✓ should set price feed, fail zero address
✓ should set price feed, invalid feed (1544ms)
✓ should set price feed should work (1396ms)
✓ should call get price should fail old price (967ms)

378 passing (9m)
```

We are grateful to have been given the opportunity to work with the 2PI team.

The statements made in this document should not be interpreted as investment or legal advice, nor should its authors be held accountable for decisions made based on them.

Zokyo's Security Team recommends that the 2PI team put in place a bug bounty program to encourage further analysis of the smart contract by third parties.

ZOKYO.