
# SECURITY AUDIT

ZOKYO.

September 8th, 2021 | v. 2.0

## PASS

Zokyo's Security Team has concluded that this smart contract passes security qualifications to be listed on digital asset exchanges.

# TECHNICAL SUMMARY

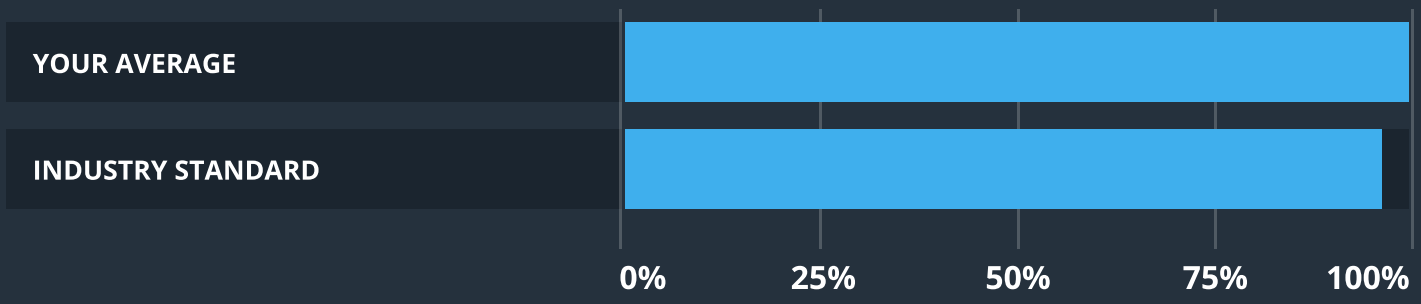
This document outlines the overall security of the Basilisk smart contracts, evaluated by Zokyo's Blockchain Security team.

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

## Contract Status


There were some critical and high issues found during the audit.

## Testable Code




The testable code is sufficient for the security, which corresponds to the industry standard.

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 security of the contract we at Zokyo recommend that the Basilisk 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 Basilisk smart contract's source code was taken from the repositories provided by the Basilisk team:

Initial commits (audited): https://github.com/galacticcouncil/Basilisk-node/releases/tag/v3.0.0
https://github.com/galacticcouncil/HydraDX-math/tree/v3.3.0

Last commits (post-audit):
https://github.com/galacticcouncil/Basilisk-node/tree/61408106039517235b699c8e5520baf89b03e93c
https://github.com/galacticcouncil/HydraDX-math/tree/846c68d99f9661e0b12b99ff21822c3ff336c8ba

Within the scope of this audit Zokyo auditors have reviewed the following contract(s):

- ./pallets/exchange/src/tests.rs
- ./pallets/lbp/src/tests.rs
- ./pallets/xyk/src/tests.rs
- ./runtime/src/lib.rs
- ./pallets/lbp/src/lib.rs
- ./pallets/exchange/src/lib.rs
- ./pallets/xyk/src/lib.rs
- ./node/src/chain_spec.rs
- ./pallets/nft/src/tests.rs
- ./node/src/command.rs
- ./pallets/transaction-multi-payment/src/tests.rs
- ./node/src/service.rs
- ./pallets/transaction-multi-payment/src/lib.rs
- ./pallets/exchange/benchmarking/src/lib.rs
- ./pallets/transaction-multi-payment/src/mock.rs
- ./pallets/lbp/src/benchmarking.rs
- ./pallets/nft/src/lib.rs
- ./pallets/exchange/src/direct.rs
- ./pallets/transaction-multi-payment/benchmarking/src/mock.rs
- ./pallets/exchange/benchmarking/src/mock.rs
- ./pallets/exchange/src/mock.rs
- ./pallets/xyk/src/mock.rs
- ./pallets/lbp/src/mock.rs
- ./pallets/exchange/src/weights.rs

- ./pallets/nft/src/benchmarking.rs
- ./pallets/nft/src/mock.rs
- ./pallets/transaction-multi-payment/benchmarking/src/lib.rs
- ./pallets/nft/src/weights.rs
- ./primitives/src/lib.rs
- ./pallets/lbp/src/weights.rs
- ./pallets/xyk/src/benchmarking.rs
- ./pallets/asset-registry/src/mock.rs
- ./node/src/rpc.rs
- ./runtime/src/weights/payment.rs
- ./runtime/src/weights/system.rs
- ./pallets/xyk/rpc/runtime-api/src/lib.rs
- ./runtime/src/currency.rs
- ./runtime/src/weights/xyk.rs
- ./primitives/src/asset.rs
- ./runtime/src/weights/timestamp.rs
- ./pallets/asset-registry/src/tests.rs
- ./node/src/main.rs
- ./pallets/transaction-multi-payment/src/traits.rs
- ./runtime/build.rs
- ./runtime/src/weights/mod.rs
- ./node/build.rs
- ./node/src/lib.rs
- ./pallets/duster/lib.rs
- ./pallets/duster/mock.rs
- ./pallets/duster/tests.rs
- ./runtime/src/tests.rs
- ./pallets/transaction-multi-payment/src/weights.rs
- ./pallets/asset-registry/src/lib.rs
- ./node/src/cli.rs
- ./pallets/xyk/rpc/src/lib.rs
- ./pallets/xyk/src/weights.rs
- ./runtime/src/weights/exchange.rs
- ./pallets/exchange/benchmarking/src/amounts.rs
- ./primitives/src/traits.rs

**HydraDX-math**

- ./src/lbp/tests.rs
- ./src/transcendental.rs
- ./src/p12.rs
- ./src/lbp/lbp.rs
- ./src/tests.rs
- ./benches/benchmarks.rs
- ./src/xyk.rs
- ./src/lib.rs
- ./src/lbp/mod.rs
- ./src/types.rs

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

- Implements and adheres to existing standards appropriately and effectively;
- Documentation and code comments match logic and behavior;
- Follows best practices in efficient use of resources, 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 Basilisk 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. 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. | 3 | Testing contract logic against common and uncommon attack vectors. |
| 2 | Cross-comparison with other, similar smart contracts by industry leaders. | 4 | Thorough, manual review of the codebase, line-by-line. |

# EXECUTIVE SUMMARY

There were several critical, high and medium issues discovered during audit, they relate to the following:

- Panic in runtime
- Extrinsic with multiple storage mutations isn’t annotated with #[transactional]
- Permission in method description doesn't match implementation
- Genesis config checks
- Integer overflow/underflow.

After recommendations by Zokyo auditors, all issues that influence security and efficiency were fixed by Basilisk.

Nevertheless, it is additionally recommended to provided several layers of fuzzy and automated testing of the functionality in order to minimize influences on the further development.

# 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 contract in such a way that funds may be lost, allocated incorrectly, or otherwise result in a significant loss.

### 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

**Extrinsic with multiple storage mutations isn’t annotated with #[transactional]**

**Occurs:**
Basilisk-node/pallets/duster/src/lib.rs:197
Basilisk-node/pallets/nft/src/lib.rs:84,107
Not using #[transactional] can lead to storage corruption.

**Recommendation:**
Use #[transactional] for every extrinsic with more than one storage mutation.

HIGH | RESOLVED

**Permission in method description doesn't match implementation**

**Occurs:**
Basilisk-node/pallets/lbp/src/lib.rs:516-568
Dispatch origin restrictions described in comments must match method implementation

**Recommendation:**
Make sure dispatch origin checks are consistent with method description.

MEDIUM | RESOLVED

### Genesis config checks

**Occurs:**
Basilisk-node/pallets/asset-registry/src/lib.rs:94-98
Absence of genesis config consistency can lead to storage corruption in case when a mistake was made in genesis config.

**Recommendation:**
Use assert!() to check that CoreAssetId, NextAssetId, AssetIds are consistent.

MEDIUM | RESOLVED

### Integer overflow/underflow

**Occurs:**
Basilisk-node/pallets/exchange/src/lib.rs:620
Basilisk-node/pallets/lbp/src/lib.rs:746,819,824
Basilisk-node/pallets/nft/src/lib.rs:98
Basilisk-node/pallets/transaction-multi-payment/src/lib.rs:367
Basilisk-node/pallets/xyk/src/lib.rs:629,752
Basilisk-node/primitives/src/lib.rs:126,132
HydraDX-math/src/lbp/lbp.rs:59,60,113
HydraDX-math/src/p12.rs:22,26,50,54,64,102
HydraDX-math/src/transcendental.rs:14,30,39,81,101
Not using safe math can lead to potential bugs.

**Recommendation:**
Use safe functions like checked_add() and checked_sub() for math operations.

LOW | UNRESOLVED

**Panic in runtime (affect further development)**

**Occurs:**
Basilisk-node/pallets/asset-registry/src/lib.rs:107
Basilisk-node/pallets/exchange/src/lib.rs:606,615,637,647
Basilisk-node/pallets/exchange/src/direct.rs:101,102,234
Basilisk-node/runtime/src/lib.rs:1107,1114,1034
HydraDX-math/src/lbp/lbp.rs:47,58,59
HydraDX-math/src/transcendental.rs:64
Usage of unwrap() or expect() functions can lead to panic.
Extrinsics must not cause a panic in the runtime logic or else the system becomes vulnerable to attacks where users can trigger computational execution without any punishment.

**Recommendation:**
Use unwrap_or(), unwrap_or_else() to provide default values or define an error in Error enum and return it explicitly when there is an error in a Result.

**Client Comment**
The client has handled or verified every case of possible panic in runtime.

**Post-audit:**
Based on the client's verification the issue is marked as low. Though since the core reason itself is not fixed, issue is left as unresolved, since it can influence the further development.

LOW | RESOLVED

### Error handling

**Occurs:**
Basilisk-node/pallets/duster/src/lib.rs:221
When dusting an account error from dec_providers() isn`t handled. This can lead to a situation when account has at least one consumer and one provider. In this situation provider counter cannot be decreased and as a result dusted account will still remain alive and subsequent dusting attempt will fail because account balance is zero.

**Recommendation:**
Revert if dec_providers() returns error.

INFORMATIONAL | UNRESOLVED

### Naming

**Occurs:**
Basilisk-node/pallets/lbp/src/lib.rs:778
Misleading method names can lead to errors during further development.

**Recommendation:**
Use names that reflect method implementation.

INFORMATIONAL | RESOLVED

### Memory allocation

**Occurs:**
Basilisk-node/primitives/src/asset.rs:54
Basilisk-node/pallets/xyk/src/lib.rs:517
Basilisk-node/runtime/src/lib.rs:1032
Extending a vector can lead to reallocation of memory and influence performance.

**Recommendation:**
Use Vec::with_capacity() to create a vector with enough capacity to store all required data.

| | asset-registry | duster | exchange |
|---|---|---|---|
| Re-entrancy | Pass | Pass | Pass |
| Access Management Hierarchy | Pass | Pass | Pass |
| Arithmetic Over/Under Flows | Pass | Pass | Pass |
| Delegatecall Unexpected Ether | Pass | Pass | Pass |
| Default Public Visibility | Pass | Pass | Pass |
| Hidden Malicious Code | Pass | Pass | Pass |
| Entropy Illusion (Lack of Randomness) | Pass | Pass | Pass |
| External Contract Referencing | Pass | Pass | Pass |
| Short Address/ Parameter Attack | Pass | Pass | Pass |
| Unchecked CALL Return Values | Pass | Pass | Pass |
| Race Conditions / Front Running | Pass | Pass | Pass |
| General Denial Of Service (DOS) | Pass | Pass | Pass |
| Uninitialized Storage Pointers | Pass | Pass | Pass |
| Floating Points and Precision | Pass | Pass | Pass |
| Tx.Origin Authentication | Pass | Pass | Pass |
| Signatures Replay | Pass | Pass | Pass |
| Pool Asset Security | Pass | Pass | Pass |

| | lbp | nft | offchain-duster |
|---|---|---|---|
| Re-entrancy | Pass | Pass | Pass |
| Access Management Hierarchy | Pass | Pass | Pass |
| Arithmetic Over/Under Flows | Pass | Pass | Pass |
| Delegatecall Unexpected Ether | Pass | Pass | Pass |
| Default Public Visibility | Pass | Pass | Pass |
| Hidden Malicious Code | Pass | Pass | Pass |
| Entropy Illusion (Lack of Randomness) | Pass | Pass | Pass |
| External Contract Referencing | Pass | Pass | Pass |
| Short Address/ Parameter Attack | Pass | Pass | Pass |
| Unchecked CALL Return Values | Pass | Pass | Pass |
| Race Conditions / Front Running | Pass | Pass | Pass |
| General Denial Of Service (DOS) | Pass | Pass | Pass |
| Uninitialized Storage Pointers | Pass | Pass | Pass |
| Floating Points and Precision | Pass | Pass | Pass |
| Tx.Origin Authentication | Pass | Pass | Pass |
| Signatures Replay | Pass | Pass | Pass |
| Pool Asset Security | Pass | Pass | Pass |

| | transaction-multipayment | xyk | math |
|---|---|---|---|
| Re-entrancy | Pass | Pass | Pass |
| Access Management Hierarchy | Pass | Pass | Pass |
| Arithmetic Over/Under Flows | Pass | Pass | Pass |
| Delegatecall Unexpected Ether | Pass | Pass | Pass |
| Default Public Visibility | Pass | Pass | Pass |
| Hidden Malicious Code | Pass | Pass | Pass |
| Entropy Illusion (Lack of Randomness) | Pass | Pass | Pass |
| External Contract Referencing | Pass | Pass | Pass |
| Short Address/ Parameter Attack | Pass | Pass | Pass |
| Unchecked CALL Return Values | Pass | Pass | Pass |
| Race Conditions / Front Running | Pass | Pass | Pass |
| General Denial Of Service (DOS) | Pass | Pass | Pass |
| Uninitialized Storage Pointers | Pass | Pass | Pass |
| Floating Points and Precision | Pass | Pass | Pass |
| Tx.Origin Authentication | Pass | Pass | Pass |
| Signatures Replay | Pass | Pass | Pass |
| Pool Asset Security | Pass | Pass | Pass |

# CODE COVERAGE AND TEST RESULTS FOR ALL FILES

## Tests written by Zokyo Secured team

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

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

**transaction-multi-payment (0.06s)**

- ✓ test mock::__construct_runtime_integrity_test::runtime_integrity_tests
- ✓ test tests::check_balance_extension_fails
- ✓ test tests::fee_payment_in_native_currency
- ✓ test tests::add_member
- ✓ test tests::add_new_accepted_currency
- ✓ test tests::check_balance_extension_works
- ✓ test tests::check_balance_should_work
- ✓ test tests::account_currency_works
- ✓ test tests::fee_payment_in_native_currency_with_no_balance
- ✓ test tests::removed_accepted_currency
- ✓ test tests::set_currency_with_insufficient_balance
- ✓ test tests::set_native_currency_with_no_balance
- ✓ test tests::fee_payment_in_non_native_currency_with_no_pool
- ✓ test tests::fee_payment_non_native_insufficient_balance_with_no_pool
- ✓ test tests::set_native_currency
- ✓ test tests::fee_payment_in_non_native_currency
- ✓ test tests::set_supported_currency_without_pool
- ✓ test tests::set_unsupported_currency
- ✓ test tests::swap_should_not_work
- ✓ test tests::weight_to_fee_should_work
- ✓ test tests::set_supported_currency_with_no_balance
- ✓ test tests::fee_payment_non_native_insufficient_balance
- ✓ test tests::set_supported_currency_with_pool
- ✓ test tests::swap_currency_should_work
- ✓ test tests::withdraw_set_fee_should_work
- ✓ test tests::swap_should_work
- ✓ test tests::withdraw_set_fee_with_core_asset_should_work

**asset-registry (0.001s)**

- ✓ test mock::__construct_runtime_integrity_test::runtime_integrity_tests
- ✓ test tests::create_asset
- ✓ test tests::create_asset_with_max_next_asset_id_should_fail

**nft (0.07s)**

- ✓ test mock::__construct_runtime_integrity_test::runtime_integrity_tests
- ✓ test tests::buy_from_pool_works
- ✓ test tests::create_pool_fails
- ✓ test tests::create_class_fails
- ✓ test tests::create_class_works
- ✓ test tests::burn_works
- ✓ test tests::buy_from_pool_fails_notapool
- ✓ test tests::burn_fails
- ✓ test tests::destroy_class_fails
- ✓ test tests::destroy_class_works
- ✓ test tests::is_owner_works
- ✓ test tests::destroy_pool_works
- ✓ test tests::create_pool_works
- ✓ test tests::is_locked_works
- ✓ test tests::destroy_pool_fails
- ✓ test tests::toggle_lock_works
- ✓ test tests::buy_from_pool_fails
- ✓ test tests::mint_works
- ✓ test tests::toggle_lock_fails
- ✓ test tests::transfer_works
- ✓ test tests::sell_to_pool_works
- ✓ test tests::mint_fails
- ✓ test tests::transfer_fails
- ✓ test tests::sell_to_pool_fails

**xyk (0.12s)**

- ✓ test mock::__construct_runtime_integrity_test::runtime_integrity_tests
- ✓ test tests::add_liquidity_to_non_existing_pool_should_not_work
- ✓ test tests::add_liquidity_should_work
- ✓ test tests::buy_with_low_amount_should_not_work
- ✓ test tests::buy_test_exceeding_max_limit
- ✓ test tests::add_liquidity_as_another_user_should_work
- ✓ test tests::buy_with_non_existing_pool_should_not_work
- ✓ test tests::buy_with_excesive_amount_should_not_work
- ✓ test tests::add_liquidity_more_than_owner_should_not_work
- ✓ test tests::add_insufficient_liquidity_should_not_work

✓ test tests::add_liquidity_exceeding_max_limit_should_not_work
✓ test tests::create_pool_overflowing_amount_should_not_work
✓ test tests::buy_without_sufficient_balance_should_not_work
✓ test tests::create_pool_with_same_assets_should_not_be_allowed
✓ test tests::create_pool_fixed_point_amount_should_work
✓ test tests::buy_without_sufficient_discount_balance_should_not_work
✓ test tests::create_pool_with_insufficient_liquidity_should_not_work
✓ test tests::destroy_pool_on_remove_liquidity_and_recreate_should_work
✓ test tests::create_pool_with_insufficient_balance_should_not_work
✓ test tests::create_pool_small_fixed_point_amount_should_work
✓ test tests::create_pool_should_work
✓ test tests::discount_buy_with_no_native_pool_should_not_work
✓ test tests::create_same_pool_should_not_work
✓ test tests::discount_sell_fees_should_work
✓ test tests::money_in_money_out_should_leave_the_same_balance_for_both_accounts
✓ test tests::fee_calculation
✓ test tests::discount_sell_with_no_native_pool_should_not_work
✓ test tests::remove_liquidity_should_work
✓ test tests::money_in_sell_money_out_should_leave_the_same_balance
✓ test tests::remove_zero_liquidity_from_non_existing_pool_should_not_work
✓ test tests::remove_zero_liquidity_should_not_work
✓ test tests::sell_test_not_reaching_limit
✓ test tests::sell_test
✓ test tests::remove_liquidity_should_respect_min_pool_limit
✓ test tests::sell_with_low_amount_should_not_work
✓ test tests::sell_with_non_existing_pool_should_not_work
✓ test tests::remove_liquidity_without_shares_should_not_work
✓ test tests::single_buy_more_than_ratio_out_should_not_work
✓ test tests::remove_liquidity_from_reduced_pool_should_not_work
✓ test tests::sell_with_correct_fees_should_work
✓ test tests::test_calculate_in_given_out
✓ test tests::single_buy_should_work
✓ test tests::sell_without_sufficient_balance_should_not_work
✓ test tests::test_calculate_out_given_in
✓ test tests::test_calculate_out_given_in_invalid
✓ test tests::test_calculate_in_given_out_insufficient_pool_balance
✓ test tests::sell_without_sufficient_discount_balance_should_not_work
✓ test tests::single_sell_more_than_ratio_in_should_not_work
✓ test tests::single_buy_with_discount_should_work
✓ test tests::work_flow_happy_path_should_work

**lbp (0.43)**

- ✓ test mock::__construct_runtime_integrity_test::runtime_integrity_tests
- ✓ test tests::amm_trait_should_work
- ✓ test tests::add_liquidity_to_non_existing_pool_should_not_work
- ✓ test tests::buy_from_non_existing_pool_should_not_work
- ✓ test tests::add_zero_liquidity_should_not_work
- ✓ test tests::add_liquidity_should_work
- ✓ test tests::add_liquidity_after_sale_started_should_work
- ✓ test tests::add_liquidity_by_non_owner_should_not_work
- ✓ test tests::calculate_weights_should_work
- ✓ test tests::add_liquidity_with_insufficient_balance_should_not_work
- ✓ test tests::calculate_fees_should_work
- ✓ test tests::create_pool_from_basic_origin_should_not_work
- ✓ test tests::buy_zero_amount_should_not_work
- ✓ test tests::buy_with_insufficient_balance_should_not_work
- ✓ test tests::create_pool_with_same_assets_should_not_work
- ✓ test tests::create_pool_with_insufficient_liquidity_should_not_work
- ✓ test tests::create_pool_with_invalid_data_should_not_work
- ✓ test tests::buy_should_work
- ✓ test tests::create_pool_with_insufficient_balance_should_not_work
- ✓ test tests::create_pool_should_work
- ✓ test tests::execute_buy_should_work
- ✓ test tests::execute_sell_should_not_work
- ✓ test tests::execute_buy_should_not_work
- ✓ test tests::create_same_pool_should_not_work
- ✓ test tests::exceed_max_out_ratio_should_not_work
- ✓ test tests::get_spot_price_should_work
- ✓ test tests::execute_sell_should_work
- ✓ test tests::invalid_fee_should_not_work
- ✓ test tests::execute_trade_should_work
- ✓ test tests::get_weights_in_order_should_work
- ✓ test tests::pause_non_existing_pool_should_not_work
- ✓ test tests::exceed_max_in_ratio_should_not_work
- ✓ test tests::exceed_trader_limit_should_not_work
- ✓ test tests::pause_pool_by_non_owner_should_not_work
- ✓ test tests::execute_trade_should_not_work
- ✓ test tests::pause_pool_should_work
- ✓ test tests::pause_non_pausable_pool_should_not_work
- ✓ test tests::remove_liquidity_from_non_existing_pool_should_not_work
- ✓ test tests::remove_liquidity_from_finalized_pool_should_work
- ✓ test tests::remove_liquidity_by_non_owner_should_not_work

✓ test tests::pause_paused_pool_should_not_work
✓ test tests::remove_liquidity_from_not_finalized_pool_should_not_work
✓ test tests::remove_liquidity_from_not_started_pool_should_work
✓ test tests::remove_liquidity_should_work
✓ test tests::sell_to_non_existing_pool_should_not_work
✓ test tests::sell_should_work
✓ test tests::pause_non_running_pool_should_not_work
✓ test tests::update_non_existing_pool_data_should_not_work
✓ test tests::remove_liquidity_from_paused_pool_should_work
✓ test tests::unpause_pool_should_work
✓ test tests::sell_zero_amount_should_not_work
✓ test tests::sell_with_insufficient_balance_should_not_work
✓ test tests::update_pool_data_should_work
✓ test tests::update_pool_data_for_running_lbp_should_not_work
✓ test tests::unpause_pool_should_not_work
✓ test tests::update_pool_interval_should_work
✓ test tests::update_pool_data_with_wrong_asset_should_not_work
✓ test tests::validate_pool_data_should_work
✓ test tests::update_pool_data_without_changes_should_not_work
✓ test tests::update_weights_and_validate_trade_should_not_work
✓ test tests::trade_in_non_running_pool_should_not_work
✓ test tests::update_pool_with_invalid_data_should_not_work
✓ test tests::zero_fee_should_work
✓ test tests::update_weights_and_validate_trade_should_work
✓ test tests::weight_update_should_work
✓ test tests::update_pool_data_by_non_owner_should_not_work
✓ test tests::zero_weight_should_not_work
✓ test tests::simulate_lbp_event_should_work

**duster (0.13s)**

✓ test mock::__construct_runtime_integrity_test::runtime_integrity_tests
✓ test tests::dust_nonexisting_account_fails
✓ test tests::dust_account_works
✓ test tests::dust_treasury_account_fails
✓ test tests::add_nondustable_account_works
✓ test tests::dust_account_with_exact_dust_fails
✓ test tests::dust_account_with_sufficient_balance_fails
✓ test tests::dust_account_native_works
✓ test tests::reward_duster_can_fail
✓ test tests::remove_nondustable_account_works
✓ test tests::native_existential_deposit

**offchain-duster (2.01s)**

- ✓ test mock::__construct_runtime_integrity_test::runtime_integrity_tests
- ✓ test tests::dust_account_works
- ✓ test tests::offchain_duster_without_dustable_accounts_does_nothing
- ✓ test tests::offchain_duster_works
- ✓ test tests::offchain_duster_dust_different_currencies
- ✓ test tests::offchain_duster_dust_several

**exchange (0.2s)**

- ✓ test mock::__construct_runtime_integrity_test::runtime_integrity_tests
- ✓ test tests::buy_trade_limits_respected_for_matched_intention
- ✓ test tests::buy_test_group_buys
- ✓ test tests::exact_match_limit_should_work
- ✓ test tests::buy_test_exact_match
- ✓ test tests::discount_tests_with_error
- ✓ test tests::discount_tests_with_discount
- ✓ test tests::in_out_calculations_error_should_work
- ✓ test tests::discount_tests_no_discount
- ✓ test tests::correct_matching
- ✓ test tests::invalid_transfers_in_resolver_should_not_work
- ✓ test tests::matching_limit_scenario_2
- ✓ test tests::matching_limits_sell_buy_should_work
- ✓ test tests::no_intentions_should_work
- ✓ test tests::matching_limit_scenario_3
- ✓ test tests::matching_limits_buy_buy_should_work
- ✓ test tests::main_intention_greater_than_matched_should_work
- ✓ test tests::sell_more_than_owner_should_not_work
- ✓ test tests::process_invalid_intention_should_work
- ✓ test tests::revert_invalid_direct_trades_should_work
- ✓ test tests::sell_test_exact_match
- ✓ test tests::sell_test_group_sells
- ✓ test tests::sell_test_inverse_standard
- ✓ test tests::sell_test_single_dot_sells
- ✓ test tests::sell_test_pool_finalization_states
- ✓ test tests::sell_trade_limits_respected_for_matched_intention
- ✓ test tests::sell_test_standard
- ✓ test tests::sell_test_mixed_buy_sells
- ✓ test tests::sell_test_single_eth_sells
- ✓ test tests::simple_buy_sell
- ✓ test tests::simple_sell_buy
- ✓ test tests::simple_buy_buy
- ✓ test tests::simple_sell_sell

✓ test tests::trade_limits_in_exact_match_scenario_should_work
✓ test tests::single_buy_intention_test
✓ test tests::sell_test_single_multiple_sells
✓ est tests::trade_limit_test
✓ test tests::single_sell_intention_test
✓ test tests::trade_min_limit
✓ test tests::trades_without_pool_should_not_work
✓ test tests::simple_sell_sell_with_error_should_not_pass
✓ test tests::register_intention_should_work
✓ test tests::register_intention_should_return_error_on_overflow
✓ test tests::execute_amm_transfer_should_work

**HydraDX-math (0.001s)**

✓ test conversion_tests::test_conversion
✓ test lbp::lbp::convert_from_fixed_should_work
✓ test lbp::lbp::convert_to_fixed_should_work
✓ test lbp::tests::linear_weights_should_work
✓ test lbp::tests::in_given_out_should_work
✓ test lbp::tests::spot_price_should_work
✓ test lbp::tests::out_given_in_should_work
✓ test tests::spot_price_should_work
✓ test tests::add_liquidity_should_work
✓ test tests::in_given_out_should_work
✓ test tests::out_given_in_should_work
✓ test tests::remove_liquidity_should_work
✓ test transcendental::tests::log2_inner_works
✓ test transcendental::tests::ln_works
✓ test transcendental::tests::powi_works
✓ test transcendental::tests::log2_works
✓ test transcendental::tests::rs_works
✓ test transcendental::tests::pow_works
✓ test transcendental::tests::exp_works

| MODULE | % BRANCH | % FUNCS | % LINES |
|---|---|---|---|
| HydraDX-math | 93.24 | 60.16 | 98.05 |
| transaction-multi-payment | 100.00 | 56.76 | 74.03 |
| asset-registry | 100.00 | 69.23 | 77.64 |
| nft | 91.18 | 63.89 | 72.41 |
| xyk | 66.67 | 53.85 | 75.1 |
| lbp | 100.00 | 65.48 | 87.19 |
| duster | 100.00 | 73.91 | 82.52 |
| offchain-duster | 100.00 | 39.32 | 93.09 |
| exchange | 100.00 | 74.19 | 95.62 |

Code coverage is stated as sufficient for the security of the contract based on the analyzied functionality, analysis of edge cases, analysis of possible incomes and based on the results of manual testing.

We are grateful to have been given the opportunity to work with the Basilisk.

**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 Basiliks put in place a bug bounty program to encourage further analysis of the smart contract by third parties.

ZOKYO.