SLOWMIST

Smart Contract Security Audit Report

Table Of Contents

1 Executive Summary

On 2025.07.29, the SlowMist security team received the Yeap-finance team's security audit application for yeap-finance, developed the audit plan according to the agreement of both parties and the characteristics of the project, and finally issued the security audit report.

The SlowMist security team adopts the strategy of "white box lead, black, grey box assists" to conduct a complete security test on the project in the way closest to the real attack.

The test method information:

Test method	Description
Black box testing	Conduct security tests from an attacker's perspective externally.
Grey box testing	Conduct security testing on code modules through the scripting tool, observing the internal running status, mining weaknesses.
White box testing	Based on the open source code, non-open source code, to detect whether there are vulnerabilities in programs such as nodes, SDK, etc.

The vulnerability severity level information:

Level	Description
Critical	Critical severity vulnerabilities will have a significant impact on the security of the DeFi project, and it is strongly recommended to fix the critical vulnerabilities.
High	High severity vulnerabilities will affect the normal operation of the DeFi project. It is strongly recommended to fix high-risk vulnerabilities.
Medium	Medium severity vulnerability will affect the operation of the DeFi project. It is recommended to fix medium-risk vulnerabilities.
Low	Low severity vulnerabilities may affect the operation of the DeFi project in certain scenarios. It is suggested that the project team should evaluate and consider whether these vulnerabilities need to be fixed.
Weakness	There are safety risks theoretically, but it is extremely difficult to reproduce in engineering.
Suggestion	There are better practices for coding or architecture.

2 Audit Methodology

The security audit process of SlowMist security team for smart contract includes two steps:

Smart contract codes are scanned/tested for commonly known and more specific vulnerabilities using automated analysis tools.

Manual audit of the codes for security issues. The contracts are manually analyzed to look for any potential problems.

Following is the list of commonly known vulnerabilities that was considered during the audit of the smart contract:

Serial Number	Audit Class	Audit Subclass
1	Overflow Audit	-
2	Replay Attack Audit	-
3	Flashloan Attack Audit	-
4	Denial of Service Audit	-
5	Race Conditions Audit	Reordering Attack Audit
6	Permission Vulnerability Audit	Access Control Audit
		Excessive Authority Audit

Serial Number	Audit Class	Audit Subclass
7	Security Design Audit	External Module Safe Use Audit
		Show Coding Security Audit
		Block data Dependence Security Audit
		Explicit Visibility of Functions Audit
8	Gas Optimization Audit	-
9	Design Logic Audit	-

Serial Number	Audit Class	Audit Subclass
10	Arithmetic Accuracy Deviation Audit	-
11	Capability Security Usage Audit	-
12	Resource Security Usage Audit	-

3 Project Overview

3.1 Project Introduction

Yeap Finance is a comprehensive decentralized finance (DeFi) platform built on the Aptos blockchain. It provides a modular and extensible architecture for lending, borrowing, yield farming, and advanced DeFi strategies like leveraged LP positions.

3.2 Vulnerability Information

The following is the status of the vulnerabilities found in this audit:

NO	Title	Category	Level	Status
N1	Unauthorized initialization function	Access Control Audit	Low	Fixed
N2	Identical log events	Others	Low	Fixed
N3	Potential risk of manipulation of hyperion llp positions	External Module Safe Use Audit	High	Fixed
N4	Slippage not checked when withdrawing llp collateral	Others	Information	Acknowledged
N5	Potential Denial of Service Risk of DAP Module	Denial of Service Audit	Low	Acknowledged
N6	Potential flaws in pyth price acquisition	Design Logic Audit	Medium	Fixed

NO	Title	Category	Level	Status
N7	Excessive Privilege Concentration	Excessive Authority Audit	Medium	Acknowledged
N8	Interest rates not updated before liquidation	Design Logic Audit	Low	Acknowledged
N9	When socialize debt is enabled, bad debt will be borne by LP	Others	Information	Acknowledged
N10	Optimizable skim operations	Design Logic Audit	Suggestion	Fixed
N11	Potential overflow risk in interest rate updates	Overflow Audit	Low	Fixed

4 Code Overview

4.1 Contracts Description

https://github.com/yeap-finance/yeap-finance

Initial audit commit: `eb6ae0aab8d44c0c7d7e41e9896923f81246c8a9`

Final review commit: `558e36f1f93485f0251b7020690d17be44325759`

```
.
├── yeap-borrow-api
│   └── sources
│       └── borrow_api.move
├── yeap-borrow-protocol-common
│   └── sources
│       ├── claimable_token_config.move
│       ├── liquidation_utils.move
│       └── risk_parameter_config.move
├── yeap-earn-api
│   └── sources
│       └── earn_api.move
├── yeap-earn-protocol
│   └── sources
│       └── earn_protocol.move
├── yeap-hyperion-llp-protocol
│   └── sources
│       ├── constants.move
│       ├── events.move
│       ├── governance.move
│       ├── health_checker.move
│       ├── initializer.move
│       ├── liquidation.move
│       ├── position_reward_claimer.move
│       ├── position.move
│       ├── protocol_handle.move
│       ├── protocol.move
│       └── risk_parameters.move
├── yeap-irm
│   └── sources
│       ├── adaptive_curve_constants.move
│       ├── adaptive_curve.move
│       ├── adaptive_irm.move
│       ├── fixed_rate_irm.move
│       ├── irm_constants.move
│       ├── irm_kind.move
│       ├── kinked_irm.move
│       ├── kinked.move
│       └── utils.move
├── yeap-lens
│   └── sources
│       ├── oracle_lens.move
│       └── scmd_position_lens.move
├── yeap-oracle
│   └── sources
│       ├── constants.move
│       ├── dap.move
│       ├── fixed_price_oracle.move
│       ├── oracle_kind.move
│       ├── oracle_router.move
│       ├── oracle.move
│       ├── primary_backup_oracle.move
│       ├── pyth_oracle.move
│       ├── switchboard_oracle.move
│       └── vault_asset_oracle.move
├── yeap-scmd-protocol
│   └── sources
│       ├── events.move
│       ├── governance.move
│       ├── health_checker.move
│       ├── initializer.move
│       ├── liquidation.move
│       ├── position_reward_claimer.move
│       ├── position.move
│       ├── protocol_constants.move
│       ├── protocol_handle.move
│       ├── protocol.move
│       └── risk_parameters.move
├── yeap-utils
│   └── sources
│       ├── bit_operations.move
│       ├── common_constants.move
│       ├── fp64_ext.move
│       ├── math128_ext.move
│       └── pow10.move
└── yeap-vault
    └── sources
        ├── constants.move
        ├── debt_asset_hooks.move
        ├── irm.move
        ├── protocol_usage_tracker.move
        ├── settings
        │   ├── auto_socialize_debt_setting.move
        │   ├── emergency_setting.move
        │   ├── fee_setting.move
        │   ├── flashloan_setting.move
        │   ├── irm_setting.move
        │   ├── pause_setting.move
        │   └── protocol_cap_setting.move
        ├── vault_asset_hooks.move
        ├── vault_events.move
        ├── vault_factory.move
        ├── vault_governance_actions.move
        ├── vault_governance_protocol.move
        ├── vault_lens.move
        ├── vault_metadata.move
        ├── vault_protocol.move
        ├── vault_snapshot.move
        ├── vault_state.move
        ├── vault_utils.move
        └── vault.move
```

The main network address of the contract is as follows:

The code was not deployed to the mainnet.

4.2 Visibility Description

The SlowMist Security team analyzed the visibility of major contracts during the audit, the result as follows:

Yeap Finance		
File	Function	Visibility
borrow_api.move	add_collateral_and_borrow	public entry
borrow_api.move	add_collateral_and_borrow_more	public entry
borrow_api.move	borrow	public entry
borrow_api.move	deposit_collateral	public entry
borrow_api.move	deposit_vault_asset_as_collateral	public entry
borrow_api.move	open_position	public entry
borrow_api.move	repay	public entry
borrow_api.move	repay_and_withdraw_collateral	public entry
borrow_api.move	repay_and_withdraw_collateral_shares	public entry
borrow_api.move	try_close_position	inline
borrow_api.move	withdraw_asset	private
borrow_api.move	withdraw_collateral	public entry
borrow_api.move	withdraw_collateral_inner	private
borrow_api.move	withdraw_collateral_share	public entry
claimable_token_config.move	claimable_tokens	public
claimable_token_config.move	fee_receiver	public
claimable_token_config.move	is_token_claimable	public
claimable_token_config.move	new	public
claimable_token_config.move	protocol_fee	public
claimable_token_config.move	remove_claimable_token	public
claimable_token_config.move	set_claimable_token	public

Yeap Finance		
claimable_token_config.move	set_fee_receiver	public
liquidation_utils.move	calculate_liquidation_max_repay_amount	public
risk_parameter_config.move	default_borrow_risk_parameters	inline
risk_parameter_config.move	default_collateral_risk_parameters	inline
risk_parameter_config.move	ensure_collateral_rp_exists	inline
risk_parameter_config.move	get_borrow_risk_weight	public
risk_parameter_config.move	get_collateral_borrow_vault_max_num	public
risk_parameter_config.move	get_collateral_liquidation_bonus_bps	public
risk_parameter_config.move	get_collateral_lltv	public
risk_parameter_config.move	get_collateral_ltv	public
risk_parameter_config.move	get_collateral_oracle	public
risk_parameter_config.move	get_collateral_risk_factor	public
risk_parameter_config.move	is_borrow_supported	public
risk_parameter_config.move	is_collateral_supported	public
risk_parameter_config.move	new	public
risk_parameter_config.move	remove_borrowable_vault	public
risk_parameter_config.move	set_borrow_risk_weight	public
risk_parameter_config.move	set_collateral_borrow_vault_max_num	public
risk_parameter_config.move	set_collateral_liquidation_bonus_bps	public
risk_parameter_config.move	set_collateral_lltv	public
risk_parameter_config.move	set_collateral_ltv	public
risk_parameter_config.move	set_collateral_risk_factor	public

Yeap Finance		
risk_parameter_config.move	set_oracle_router	public
risk_parameter_config.move	validate_borrow_rps	private
risk_parameter_config.move	set_borrow_risk_weight	public
risk_parameter_config.move	remove_borrowable_vault	public
risk_parameter_config.move	get_collateral_oracle	public
risk_parameter_config.move	get_collateral_risk_factor	public
risk_parameter_config.move	get_collateral_ltv	public
risk_parameter_config.move	get_collateral_lltv	public
risk_parameter_config.move	get_collateral_liquidation_bonus_bps	public
risk_parameter_config.move	get_collateral_borrow_vault_max_num	public
risk_parameter_config.move	get_borrow_risk_weight	public
risk_parameter_config.move	is_collateral_supported	public
risk_parameter_config.move	is_borrow_supported	public
risk_parameter_config.move	validate_borrow_rps	private
earn_api.move	deposit	public entry
earn_api.move	emit_vault_user_operation_event	private
earn_api.move	redeem	public entry
earn_api.move	withdraw	public entry
earn_protocol.move	deposit	public
earn_protocol.move	redeem	public
events.move	emit_liquidation_event	friend
events.move	emit_position_add_debt_event	friend

Yeap Finance		
events.move	emit_position_bad_debt_event	friend
events.move	emit_position_collateral_updated_event	friend
events.move	emit_position_created_event	friend
events.move	emit_position_destroyed_event	friend
events.move	emit_position_remove_debt_event	friend
governance.move	generate_config_object_signer	public
governance.move	require_governance	public
health_checker.move	amount_to_value	inline
health_checker.move	check_position_healthy	friend
health_checker.move	get_or_fetch_oracle_price	private
health_checker.move	get_position_collateral_value_with_price_cache	friend
health_checker.move	get_position_loan_value_with_price_cache	friend
initializer.move	init_module	private
initializer.move	init_module_test_only	public
liquidation.move	amount_to_value	inline
liquidation.move	collateral_balance	inline
liquidation.move	debt_shares	inline
liquidation.move	liquidate	public
position.move	collateral_asset_type	public
position.move	create	friend
position.move	create_debt_store_inner	inline
position.move	debt_store	public

Yeap Finance		
position.move	debt_stores	public
position.move	destroy	friend
position.move	ensure_debt_store_exists	friend
position.move	fee_tier	public
position.move	position_signer	friend
position.move	remove_debt_store	friend
position.move	remove_debt_store_inner	inline
position.move	set_collateral	friend
position.move	token_a	public
position.move	token_b	public
position.move	underlying_position_object	public
position.move	withdraw_collateral	friend
protocol.move	borrow	public
protocol.move	clear_borrow	public
protocol.move	close_position	public
protocol.move	commit	public
protocol.move	handle_bad_debt	friend
protocol.move	open_position	public
protocol.move	position_address	public
protocol.move	protocol_address	inline
protocol.move	repay	public
protocol.move	require_position_ownership	inline

Yeap Finance		
protocol.move	transfer_in_collateral	public
protocol.move	trasfer_out_collateral	public
protocol.move	tx	public
protocol.move	withdraw_asset	friend
position_reward_claimer.move	claim	public
protocol_handle.move	protocol	friend
risk_parameters.move	get_borrow_risk_weight	public inline
risk_parameters.move	get_collateral_borrow_vault_max_num	public inline
risk_parameters.move	get_collateral_liquidation_bonus	public inline
risk_parameters.move	get_collateral_lltv	public inline
risk_parameters.move	get_collateral_ltv	public inline
risk_parameters.move	get_collateral_oracle	public inline
risk_parameters.move	get_collateral_risk_factor	public inline
risk_parameters.move	is_borrow_supported	public inline
risk_parameters.move	is_collateral_supported	public inline
adaptive_curve.move	calc_err	private
adaptive_curve.move	compute_interest_rate	public
adaptive_curve.move	curve	private
adaptive_curve.move	ensure_valid	private
adaptive_curve.move	new	public
adaptive_curve.move	new_rate_at_target	private
adaptive_irm.move	compute_interest_rate	public

Yeap Finance		
adaptive_irm.move	compute_interest_rate_inner	private
adaptive_irm.move	initial_state	inline
adaptive_irm.move	initialize	public
adaptive_irm.move	update_interest_rate	public
adaptive_irm.move	update_settings	public
fixed_rate_irm.move	compute_interest_rate	public
fixed_rate_irm.move	compute_interest_rate_inner	inline
fixed_rate_irm.move	initialize	public
fixed_rate_irm.move	update_interest_rate	public
fixed_rate_irm.move	update_settings	public
kinked.move	compute_interest_rate	public
kinked.move	ensure_valid	private
kinked.move	new	public
kinked_irm.move	compute_interest_rate	public
kinked_irm.move	compute_interest_rate_inner	inline
kinked_irm.move	initialize	public
kinked_irm.move	update_interest_rate	public
kinked_irm.move	update_settings	public
utils.move	calculate_utilization	public
oracle_lens.move	get_price	public
oracle_lens.move	get_price_of_pair	public
scmd_position_lens.move	amount_to_value	inline

Yeap Finance		
scmd_position_lens.move	calculate_health_factor	public
scmd_position_lens.move	calculate_max_borrow_capacity	public
scmd_position_lens.move	check_position_healthy	friend
scmd_position_lens.move	collateral_balance	public
scmd_position_lens.move	debt_amounts	public
scmd_position_lens.move	debt_shares	public
scmd_position_lens.move	get_or_fetch_oracle_price	private
scmd_position_lens.move	get_position_collateral_value	public
scmd_position_lens.move	get_position_collateral_value_with_price_cache	friend
scmd_position_lens.move	get_position_loan_value	public
scmd_position_lens.move	get_position_loan_value_with_price_cache	friend
scmd_position_lens.move	is_position_healthy	public
scmd_position_lens.move	is_position_liquidatable	public
scmd_position_lens.move	value_to_amount	inline
dap.move	create_test_dap	private
fixed_price_oracle.move	get_price	public
fixed_price_oracle.move	initialize	public
fixed_price_oracle.move	remove_fixed_price	public
fixed_price_oracle.move	set_fixed_price	public
oracle.move	get_quote	public
oracle_kind.move	delegate_oracle_kind	public inline
oracle_kind.move	fix_price_oracle_kind	public inline

Yeap Finance		
oracle_kind.move	primary_backup_oracle_kind	public inline
oracle_kind.move	vault_oracle_kind	public inline
oracle_router.move	compute_price	private
oracle_router.move	create	public
oracle_router.move	get_oracle_config	public
oracle_router.move	get_price	public(friend)
oracle_router.move	get_price_inner	private
oracle_router.move	set_oracle	public
oracle_router.move	unset_oracle	public
primary_backup_oracle.move	get_price	public
pyth_oracle.move	check_price	private
pyth_oracle.move	check_price_for_test	public
pyth_oracle.move	config_exists	public
pyth_oracle.move	get_price_if_valid_scaled	public
pyth_oracle.move	initialize	public
pyth_oracle.move	remove_config	public
pyth_oracle.move	set_config	public
switchboard_oracle.move	check_price	private
switchboard_oracle.move	config_exists	public
switchboard_oracle.move	get_price_if_valid_scaled	public
switchboard_oracle.move	initialize	public
switchboard_oracle.move	new_switchboard_config	inline

Yeap Finance		
switchboard_oracle.move	remove_config	public
switchboard_oracle.move	set_config	public
vault_asset_oracle.move	get_price	public
events.move	emit_liquidation_event	friend
events.move	emit_position_add_debt_event	friend
events.move	emit_position_bad_debt_event	friend
events.move	emit_position_created_event	friend
events.move	emit_position_destroyed_event	friend
events.move	emit_position_remove_debt_event	friend
governance.move	generate_config_object_signer	public
governance.move	require_governance	public
health_checker.move	amount_to_value	inline
health_checker.move	check_position_healthy	friend
health_checker.move	get_or_fetch_oracle_price	private
health_checker.move	get_position_collateral_value_with_price_cache	friend
health_checker.move	get_position_loan_value_with_price_cache	friend
initializer.move	init_module	private
liquidation.move	amount_to_value	inline
liquidation.move	collateral_balance	inline
liquidation.move	debt_shares	inline
liquidation.move	liquidate	public
position.move	collateral_asset_type	public

Yeap Finance		
position.move	collateral_store	public
position.move	create	friend
position.move	create_debt_store_inner	inline
position.move	debt_store	public
position.move	debt_stores	public
position.move	destroy	friend
position.move	ensure_debt_store_exists	friend
position.move	position_signer	friend
position.move	remove_debt_store	friend
position.move	remove_debt_store_inner	inline
position.move	withdraw_asset	friend
position.move	withdraw_collateral	friend
position_reward_claimer.move	claim	public
protocol.move	borrow	public
protocol.move	clear_borrow	public
protocol.move	close_position	public
protocol.move	commit	public
protocol.move	handle_bad_debt	friend
protocol.move	open_position	public
protocol.move	position_address	public
protocol.move	repay	public
protocol.move	require_position_ownership	inline

Yeap Finance		
protocol.move	tx	public
protocol.move	withdraw_collateral	public
protocol_constants.move	config_address	public inline
protocol_constants.move	protocol_address	public inline
protocol_constants.move	protocol_config_seed	public inline
protocol_handle.move	protocol	public(friend)
risk_parameters.move	get_borrow_risk_weight	public inline
risk_parameters.move	get_collateral_borrow_vault_max_num	public inline
risk_parameters.move	get_collateral_liquidation_bonus	public inline
risk_parameters.move	get_collateral_lltv	public inline
risk_parameters.move	get_collateral_ltv	public inline
risk_parameters.move	get_collateral_oracle	public inline
risk_parameters.move	get_collateral_risk_factor	public inline
risk_parameters.move	is_borrow_supported	public inline
risk_parameters.move	is_collateral_supported	public inline
bit_operations.move	clear_bit	public
bit_operations.move	is_bit_set	public
bit_operations.move	set_bit	public
fp64_ext.move	div_fp64	public
fp64_ext.move	mul_fp64	public
math128_ext.move	mul_div_roundup	public
math128_ext.move	pow	public

Yeap Finance		
pow10.move	pow10	public
constants.move	debt_asset_object_seed	public inline
constants.move	interest_fee_scale	public inline
constants.move	vault_config_object_seed	public inline
irm.move	update_interest_rate	friend
protocol_usage_tracker.move	initialize	friend
protocol_usage_tracker.move	get_protocol_borrow	public
protocol_usage_tracker.move	track_borrow	friend
protocol_usage_tracker.move	track_repay	friend
vault.move	flashloan	public
vault.move	payback_flashloan	public
vault.move	repay_bad_debt	public
vault.move	touch	public
vault.move	deposit	public
vault.move	redeem	public
vault.move	borrow	public
vault.move	mark_bad_debt	public
vault.move	repay	public
vault_events.move	emit_bad_debt_marked_event	friend
vault_events.move	emit_bad_debt_repay_event	friend
vault_events.move	emit_bad_debt_socialized_event	friend
vault_events.move	emit_flashloan_event	friend

Yeap Finance		
vault_events.move	emit_vault_created_event	friend
vault_events.move	emit_vault_state_change_event	friend
vault_factory.move	create	public
vault_governance_actions.move	emergency_withdraw	public
vault_governance_actions.move	skim	public
vault_governance_actions.move	socialize_bad_debt	public
vault_governance_actions.move	sync	public
vault_governance_protocol.move	assert_admin	private
vault_governance_protocol.move	assert_vault_config_object_signer	public
vault_governance_protocol.move	generate_config_object_signer	public
vault_governance_protocol.move	protocol	private
vault_lens.move	latest_state	public
vault_lens.move	preview_borrow	public
vault_lens.move	preview_deposit	public
vault_lens.move	preview_mint	public
vault_lens.move	preview_redeem	public
vault_lens.move	preview_repay	public
vault_lens.move	preview_repay_shares	public
vault_lens.move	preview_total_bad_debt	public
vault_lens.move	preview_total_borrows	public
vault_lens.move	preview_total_cash	public
vault_lens.move	preview_total_debt_shares	public

Yeap Finance		
vault_lens.move	preview_total_shares	public
vault_lens.move	preview_withdraw	public
vault_lens.move	state_stored	public
vault_metadata.move	assert_is_vault	public
vault_metadata.move	config_object	public inline
vault_metadata.move	debt_metadata	public inline
vault_metadata.move	initialize	friend
vault_metadata.move	underlying_asset_metadata	public inline
vault_metadata.move	underlying_asset_store	public
vault_metadata.move	vault_metadata	public inline
vault_protocol.move	protocol	friend
vault_snapshot.move	bad_debt	public
vault_snapshot.move	borrows	public
vault_snapshot.move	cash	public
vault_snapshot.move	last_interest_accumulator_update_time	public
vault_snapshot.move	new	friend
vault_snapshot.move	preview_accure_interest	public
vault_snapshot.move	preview_borrow	public
vault_snapshot.move	preview_deposit	public
vault_snapshot.move	preview_mint	public
vault_snapshot.move	preview_redeem	public
vault_snapshot.move	preview_repay	public

Yeap Finance		
vault_snapshot.move	preview_repay_shares	public
vault_snapshot.move	preview_withdraw	public
vault_snapshot.move	total_debt_shares	public
vault_snapshot.move	total_shares	public
vault_state.move	advance_state	friend
vault_state.move	decrease_bad_debt	friend
vault_state.move	decrease_borrows	friend
vault_state.move	deposit_underlying_asset	friend
vault_state.move	directly_withdraw_underlying_asset	friend
vault_state.move	emit_vault_state_event	friend
vault_state.move	increase_bad_debt	friend
vault_state.move	increase_borrows	friend
vault_state.move	initialize	friend
vault_state.move	latest_state	friend
vault_state.move	snapshot	inline
vault_state.move	state_stored	friend
vault_state.move	sync	friend
vault_state.move	update_interest_rate	friend
vault_state.move	withdraw_underlying_asset	friend
vault_utils.move	asset_to_share	public
vault_utils.move	asset_to_share_then_round_up_to_asset	private
vault_utils.move	share_to_asset	public

Yeap Finance		
vault_utils.move	share_to_asset_round_up_then_round_downto_share	private

4.3 Vulnerability Summary

[N1] [Low] Unauthorized initialization function

Category: Access Control Audit

Content

The Yeap Finance protocol has multiple critical initialization functions that lack proper access control mechanisms, allowing any user to call these functions for system initialization.

Code location:

- yeap-borrow-protocol-common/sources/claimable_token_config.move

```
public fun new(caller: &signer) {
   move_to(caller, ClaimableTokenConfig { /* ... */ });
}
```

- yeap-borrow-protocol-common/sources/risk_parameter_config.move

```
public fun new(config_signer: &signer) {
    move_to(config_signer, CollateralRiskParametersConfig { /* ... */ });
    move_to(config_signer, BorrowRiskParametersConfig { /* ... */ });
}
```

- yeap-oracle/sources/fixed_price_oracle.move

```
public fun initialize(account: &signer) {
    move_to(account, FixedPriceConfigs { /* ... */ });
}
```

- yeap-oracle/sources/oracle_router.move

```
public fun create(account: &signer) {
    move_to(account, OracleRouter { /* ... */ });
}
```

- yeap-oracle/sources/pyth_oracle.move

```
public fun initialize(account: &signer) {
    move_to(account, PythOracle { /* ... */ });
}
```

- yeap-oracle/sources/switchboard_oracle.move

```
public fun initialize(account: &signer) {
    move_to(account, SwitchboardOracle { /* ... */ });
}
```

- yeap-irm/sources/adaptive_irm.move

```
public fun initialize(signer: &signer, /* params */) {
    move_to(signer, AdaptiveIRM { /* ... */ });
}
```

- yeap-irm/sources/fixed_rate_irm.move

```
public fun initialize(signer: &signer, /* params */) {
    move_to(signer, FixedRateIRM { /* ... */ });
}
```

- yeap-irm/sources/kinked_irm.move

```
public fun initialize(signer: &signer, /* params */) {
    move_to(signer, KinkedIRM { /* ... */ });
}
```

Attackers may control configuration objects to set malicious parameters.

Solution

Restrict initialization permissions.

Status

Fixed; Partial fixed in new design of borrow market.

[N2] [Low] Identical log events

Category: Others

Content

There are multiple instances of identical event emission issues in the Yeap Finance protocol, which can make it difficult for off-chain systems to distinguish the operation functions that trigger the events, potentially leading to errors in off-chain business systems.

- yeap-earn-api/sources/earn_api.move

```
  81,9:         emit_vault_user_operation_event(address_of(user), vault,
string::utf8(b"withdraw"), fungible_asset::amount(&underlyings), shares);
  112,9:         emit_vault_user_operation_event(address_of(user), vault,
string::utf8(b"withdraw"), underlying_amount, shares_to_burn)
```

- yeap-vault/sources/vault_governance_actions.move

```
  72,22:         vault_state::emit_vault_state_event(vault_address);
  110,22:         vault_state::emit_vault_state_event(vault_address);
  150,22:         vault_state::emit_vault_state_event(vault_address);
```

- yeap-vault/sources/vault.move

```
  82,9:         emit_vault_state_event(vault_address);
  147,9:         emit_vault_state_event(vault_address);
  210,9:         emit_vault_state_event(vault_address);
  307,9:         emit_vault_state_event(vault_address);
  417,9:         emit_vault_state_event(vault_address);
  466,9:         emit_vault_state_event(vault_address);
  565,9:         emit_vault_state_event(vault_address);
```

Solution

Customize different events for each call.

Status

Fixed

[N3] [High] Potential risk of manipulation of hyperion llp positions

Category: External Module Safe Use Audit

Content

In the `health_checker` module of the `yeap_hyperion_llp_protocol`, the `get_position_collateral_value_with_price_cache` function is used to calculate the collateral value of a user's leveraged LP position. Within this function, the `get_amount_by_liquidity` interface of Router V3 is utilized to obtain the token amounts corresponding to the user's LP position. It is crucial to note that if the calculation method of `get_amount_by_liquidity` is tied to the real-time price of the pool, then the token amounts calculated through this interface can be manipulated. A malicious user could perform a large swap within the same transaction to alter the pool's price, thereby manipulating the amounts of the two tokens corresponding to the user's LP. However, the price from the external oracle would not change, leading to the manipulation of the user's collateral value. Malicious users could exploit this vulnerability to engage in over-borrowing or to maliciously liquidate other users.

Code location: yeap-hyperion-llp-protocol/sources/health_checker.move#L230

```
   friend fun get_position_collateral_value_with_price_cache(
        position: address, price_cache: &mut SimpleMap<Object<Metadata>, u128>
    ): (u128, u128) {
        ...

        let (position_token_a_amount, position_token_b_amount) =
router_v3::get_amount_by_liquidity(hyperion_position);

        ...
    }
```

Solution

It is recommended that after obtaining the quantities of the two tokens, their ratio be checked against the external oracle price to ensure that they have not been manipulated.

Status

Fixed

[N4] [Information] Slippage not checked when withdrawing llp collateral

Category: Others

Content

In the `position` module of the `yeap_hyperion_llp_protocol`, the `withdraw_collateral` function is used to remove a user's LLP position liquidity from Hyperion and transfer the acquired tokens to the user. However, when removing liquidity, the necessary `amount_a_min` and `amount_b_min` parameters are not set. This implies that MEV (Maximal Extractable Value) or other malicious users could execute a sandwich attack, leading to potential asset loss for the user.

Code location:

- yeap-hyperion-llp-protocol/sources/llposition.move#L383-L384

```
    friend fun withdraw_collateral(position_id: address, amount: u128): (FungibleAsset, FungibleAsset) acquires LLPosition {
        ...

        router_v3::remove_liquidity(
            &position_signer,
            underlying_position,
            amount,
            0,
            0,
            position_id,
            timestamp::now_seconds()
        );
        ...
    }
```

Solution

It is recommended to calculate slippage off-chain and apply it when removing liquidity, or alternatively, to return the LLP tokens to the user when they withdraw collateral, instead of the two underlying tokens after liquidity removal.

Status

Acknowledged

[N5] [Low] Potential Denial of Service Risk of DAP Module

Category: Denial of Service Audit

Content

In `yeap_oracle`, the DAP module is primarily responsible for asset price routing, constructing a directed acyclic graph (DAG) to manage price conversion paths between different assets. However, operations within functions such as `add_edge`, `get_common_ancestor`, and `get_paths` all require traversing the entire path to detect cycles. This traversal has a time complexity of O(n). If the created paths become excessively long, it could lead to a Denial-of-Service (DoS) risk.

Code location:

- yeap-oracle/sources/dap.move#L240,L420,L461

```
 friend fun add_edge<K: store + drop + copy, V: store + drop + copy>(
    self: &mut DAP<K, V>, from: K, to: K, data: V
  ) {
    ...
    loop {
      if (!self.edge_mapping.contains(current_node)) { break };
      current_node = *self.edge_mapping.borrow(current_node);
      if (current_node == from) {
        would_create_cycle = true;
        break
      };
    };

    ...
  }

  friend fun get_common_ancestor<K: store + drop + copy, V: store + drop + copy>(
    self: &DAP<K, V>, a: K, b: K
  ): option::Option<K> {
    ...
    while (i > 0 && j > 0) {
      ...
    };

    ...
  }


  friend fun get_paths<K: store + drop + copy, V: store + drop + copy>(self: &DAP<K, V>, a: K, until: K): vector<K> {
    ...
    loop {
      ancestors.push_back(current_node);
      // end if we find
```

```
            if (current_node == until) { break };
            // or else to the end
            if (!self.edge_mapping.contains(current_node)) {
                break;
            };
            current_node = *self.edge_mapping.borrow(current_node);
        };
        ancestors
    }
```

Solution

It is recommended to control the path length. When paths are excessively long, consider batch processing or employing more efficient algorithms.

Status

Acknowledged

[N6] [Medium] Potential flaws in pyth price acquisition

Category: Design Logic Audit

Content

In the `pyth_oracle` module of `yeap_oracle`, the `get_price_if_valid_scaled` function is used to retrieve asset prices from the Pyth oracle and perform necessary price checks. These checks include verifying the oracle price update interval. However, it's crucial to note that the protocol's operation relies on real-time and accurate prices, and Pyth is a "Pull Oracle." This means that if no user actively pulls and updates the price on-chain, the on-chain oracle might remain un-updated for extended periods, rendering the protocol's Pyth price source unavailable.

Code location:

- yeap-oracle/sources/pyth_oracle.move#L339-L346

```
    fun check_price(self: &PythConfig, pyth_price_feed: &Price): Option<u128> {
        ...
        let age =
            if (current_timestamp > pyth_timestamp) {
                current_timestamp - pyth_timestamp
            } else {
                pyth_timestamp - current_timestamp
            };
        if (age > self.pyth_max_age_in_seconds) {
            return option::none() // Price is stale
        };

        ...
        option::some(scaled_price)

    }
```

Solution

It is recommended that users update the Pyth oracle price concurrently with their position operations, allowing them to utilize real-time prices.

Status

Fixed

[N7] [Medium] Excessive Privilege Concentration

Category: Excessive Authority Audit

Content

Three core governance modules in the YeaP Finance protocol suffer from issues of excessive concentration of power.

A single governance entity holds nearly unlimited permissions and can arbitrarily modify key protocol parameters, lacking effective checks and balances.

- yeap-hyperion-llp-protocol/sources/governance.move

  `governance` can:

- Fully control the Hyperion LLP protocol configuration
- Modify risk parameters and liquidation settings
- Manage all protocol-level operations
- yeap-scmd-protocol/sources/governance.move

  `governance` can:

- Fully control the SCMD protocol configuration
- Set collateral and lending parameters
- Manage liquidation and risk control mechanisms
- yeap-vault/sources/vault_governance_protocol.move

  `admin` can:

- Synchronize vault status and skim assets
- Socialize bad debt processing
- Emergency asset withdrawal
- Enable/disable protocol integrations
- Modify fees and pause settings

Solution

The current over-concentration of governance power is a serious security risk, which may lead to single-point control and abuse of the protocol. It is recommended to immediately implement a multi-signature mechanism and gradually transition to a more decentralized governance model to protect user funds and maintain the credibility of the protocol.

Status

Acknowledged; When mainnet, all protocol level admin will a multisig wallet controlled by all team member using aptos native multisig functions.

[N8] [Low] Interest rates not updated before liquidation

Category: Design Logic Audit

Content

In the `liquidation` module, users can liquidate insolvent users via the `liquidate` function. However, the protocol's global interest rate is not updated prior to liquidation. This could lead to interest not being accounted for when calculating user liabilities through `preview_repay_shares`, thereby impacting the accuracy of the liquidation operation.

Code location:

- yeap-hyperion-llp-protocol/sources/liquidation.move#L201

```
public fun liquidate(
     position_id: address, repay_vault: address, repay_asset: &mut FungibleAsset
 ): (FungibleAsset, FungibleAsset) {
     ...
 }
```

- yeap-scmd-protocol/sources/scmd_liquidation.move#L159

```
public fun liquidate(
     position_id: address, repay_vault: address, repay_asset: &mut FungibleAsset
 ): FungibleAsset {
     ...
 }
```

Solution

It is recommended to perform a `touch` operation on the corresponding vault before proceeding with liquidation.

Status

Acknowledged

[N9] [Information] When socialize debt is enabled, bad debt will be borne by LP

Category: Others

Content

In the `vault` module, when a user is liquidated and becomes insolvent, the protocol marks the outstanding amount as bad debt via the `mark_bad_debt` function. If `auto_socialize_debt_setting` is enabled, this bad debt within the current vault will be cleared directly without requiring additional liquidity. This implies that the bad debt will be borne by the LPs.

Code location:

- yeap-vault/sources/vault.move#L414-L426

```
public fun mark_bad_debt<BorrowProtocol>(
      ...
  ) {
      ...

      // Check if auto-socialize is enabled and automatically socialize the bad debt
      if (auto_socialize_debt_setting::is_enabled(vault_address)) {
          let current_bad_debt = vault_snapshot_after.bad_debt();

          // Directly update bad debt field for efficiency
          vault_state::decrease_bad_debt(vault_address, asset_to_repay);
          // Emit bad debt socialized event for proper tracking
          emit_bad_debt_socialized_event(
              vault_address,
              asset_to_repay,
              current_bad_debt,
              current_bad_debt - (asset_to_repay as u128)
          );
      };
      ...
  }
```

Solution

Status

Acknowledged

[N10] [Suggestion] Optimizable skim operations

Category: Design Logic Audit

Content

In the `vault_governance_actions` module, the governance (gov) can claim unclaimed tokens within the vault via the `skim` function. This function checks whether the actual token amount held by the contract is greater than or equal to the amount recorded in `vault_snapshot`. However, it should be noted that using the `<=` comparison operator for this check is inappropriate, as performing a `skim` operation when token amounts are equal is unnecessary.

Code location:

- yeap-vault/sources/vault_governance_actions.move#L104

```
    public fun skim(governance_signer: &signer, vault_address: address): FungibleAsset
{
        ...
        assert!(cash <= actual_balance,
error::resource_exhausted(E_BALANCE_NOT_ENOUGH));

        ...
    }
```

Solution

It is recommended to use the `<` comparison operator when checking the cash value.

Status

Fixed

[N11] [Low] Potential overflow risk in interest rate updates

Category: Overflow Audit

Content

In the `vault_snapshot` module, the `preview_accure_interest` function is used to calculate the protocol's accumulated interest rate. When calculating the `multiplier`, the `pow` function is used for exponentiation. This implies that if the protocol's interest rate is not updated for an extended period, `timeElapsed` could become excessively large, causing the `multiplier` calculation to overflow and lead to an abort. This would prevent the protocol from performing normal interest rate updates, resulting in a Denial of Service (DoS).

Code location:

- yeap-vault/sources/vault_snapshot.move#L182-L187

```
public fun preview_accure_interest(
     ...
 ): VaultStateSnapshot {
     ...
     let multiplier =
         math128_ext::pow(
             interest_rate + yeap_irm::irm_constants::IR_SCALE(),
             delta_time_in_seconds as u128,
             yeap_irm::irm_constants::IR_SCALE()
         );

     // sanity check: multiplier >= 1
     assert!(
         multiplier >= (yeap_irm::irm_constants::IR_SCALE()),
         error::internal(E_IR_MULTIPLIER)
     );

     ...
 }
```

Solution

It is recommended that the project team implement an off-chain scheduled script to automatically call the `touch` function at fixed time intervals to update interest rates, thereby mitigating the aforementioned risk.

Status

Fixed

5 Audit Result

Audit Number	Audit Team	Audit Date	Audit Result
0X002508140002	SlowMist Security Team	2025.07.29 - 2025.08.14	Medium Risk

Summary conclusion: The SlowMist security team use a manual and SlowMist team's analysis tool to audit the project, during the audit work we found 1 high risk, 2 medium risk, 5 low risk, 1 suggestion vulnerabilities.

6 Statement

SlowMist issues this report with reference to the facts that have occurred or existed before the issuance of this report, and only assumes corresponding responsibility based on these.

For the facts that occurred or existed after the issuance, SlowMist is not able to judge the security status of this project, and is not responsible for them. The security audit analysis and other contents of this report are based on the documents and materials provided to SlowMist by the information provider till the date of the insurance report (referred to as "provided information"). SlowMist assumes: The information provided is not missing, tampered with, deleted or concealed. If the information provided is missing, tampered with, deleted, concealed, or inconsistent with the actual situation, the SlowMist shall not be liable for any loss or adverse effect resulting therefrom. SlowMist only conducts the agreed security audit on the security situation of the project and issues this report. SlowMist is not responsible for the background and other conditions of the project.

SLOWMIST

Official Website

www.slowmist.com

✉

E-mail

team@slowmist.com

Twitter

@SlowMist_Team

Github

https://github.com/slowmist