// Smart Contract Security Assessment 12.13.2024 - 02.28.2025

Glow
Blueprint Finance

HALBORN

Glow - Blueprint Finance

Prepared by: HALBORN

Last Updated 03/19/2025

Date of Engagement by: December 13th, 2024 - February 28th, 2025

Summary

100% ⓘ OF ALL REPORTED FINDINGS HAVE BEEN ADDRESSED

ALL FINDINGS	CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL
16	0	0	1	7	8

TABLE OF CONTENTS

1. Introduction

Blueprint Finance engaged Halborn to conduct a security assessment on their **Glow V1** and **Lookup Table Registry Solana programs** beginning on December 13th, 2024, and ending on February 28th, 2025. The security assessment was scoped to the Solana Programs provided in glow-v1 and lookup-table-registry GitHub repositories. Commit hashes and further details can be found in the Scope section of this report.

The Glow V1 program is a **noncustodial borrowing and lending protocol** consisting of four interconnected programs:

- Airspace – Implements a permission system that isolates markets and controls which programs users in those markets can interact with.
- Metadata – Manages program metadata, including enabled trading/deposit tokens and oracle information.
- Margin – Allows users to interact with other programs using lower collateral than required for direct interaction.
- Margin-Pool – Implements a variable-rate lending and borrowing pools.

The Lookup Table Registry is a program for creating and tracking address lookup tables.

During the security assessment, the Glow team implemented two code updates:

- Removed the control program previously used for bootstrapping and configuring margin pools.
- Added support for the pull-based Pyth oracle architecture.

2. Assessment Summary

Halborn was provided 11 weeks for the engagement and assigned one full-time security engineer to review the security of the Solana Programs in scope. The engineer is a blockchain and smart contract security expert with advanced smart contract hacking skills, and deep knowledge of multiple blockchain protocols.

The purpose of the assessment is to:

- Identify potential security issues within the Solana Programs.
- Ensure that smart contract functionality operates as intended.

In summary, Halborn identified some security concerns that were mostly addressed by the Glow team. The main ones were the following:

- Verify that registry accounts can be closed only by a dedicated authority.
- Implement two step authority transfer.
- Make sure the administrator cannot transfer positions to different user.
- Properly verify instruction parameters.
- Validate mint extensions to avoid potentially dangerous extensions.
- Make sure the liquidator are incentivized to restore accounts to healthy state.
- Prevent the airspace authorities to manipulate arbitrary metadata accounts.
- Optimize reallocations by ensuring new memory is not zeroed twice.

3. Test Approach And Methodology

Halborn performed a combination of a manual review of the source code and automated security testing to balance efficiency, timeliness, practicality, and accuracy in regard to the scope of the program assessment. While manual testing is recommended to uncover flaws in business logic, processes, and implementation; automated testing techniques help enhance coverage of programs and can quickly identify items that do not follow security best practices.

The following phases and associated tools were used throughout the term of the assessment:

- Research into the architecture, purpose, and use of the platform.

- Manual program source code review to identify business logic issues.

- Mapping out possible attack vectors

- Thorough assessment of safety and usage of critical Rust variables and functions in scope that could lead to arithmetic vulnerabilities.

- Scanning dependencies for known vulnerabilities (`cargo audit`).

- Local runtime testing (`solana-test-framework`)

4. RISK METHODOLOGY

Every vulnerability and issue observed by Halborn is ranked based on **two sets** of **Metrics** and a **Severity Coefficient**. This system is inspired by the industry standard Common Vulnerability Scoring System.

The two **Metric sets** are: **Exploitability** and **Impact**. **Exploitability** captures the ease and technical means by which vulnerabilities can be exploited and **Impact** describes the consequences of a successful exploit.

The **Severity Coefficients** is designed to further refine the accuracy of the ranking with two factors: **Reversibility** and **Scope**. These capture the impact of the vulnerability on the environment as well as the number of users and smart contracts affected.

The final score is a value between 0-10 rounded up to 1 decimal place and 10 corresponding to the highest security risk. This provides an objective and accurate rating of the severity of security vulnerabilities in smart contracts.

The system is designed to assist in identifying and prioritizing vulnerabilities based on their level of risk to address the most critical issues in a timely manner.

4.1 EXPLOITABILITY

ATTACK ORIGIN (AO):

Captures whether the attack requires compromising a specific account.

ATTACK COST (AC):

Captures the cost of exploiting the vulnerability incurred by the attacker relative to sending a single transaction on the relevant blockchain. Includes but is not limited to financial and computational cost.

ATTACK COMPLEXITY (AX):

Describes the conditions beyond the attacker's control that must exist in order to exploit the vulnerability. Includes but is not limited to macro situation, available third-party liquidity and regulatory challenges.

METRICS:

EXPLOITABILITY METRIC (M_E)	METRIC VALUE	NUMERICAL VALUE
Attack Origin (AO)	Arbitrary (AO:A) Specific (AO:S)	1 0.2

EXPLOITABILITY METRIC (M_E)	METRIC VALUE	NUMERICAL VALUE
Attack Cost (AC)	Low (AC:L) Medium (AC:M) High (AC:H)	1 0.67 0.33
Attack Complexity (AX)	Low (AX:L) Medium (AX:M) High (AX:H)	1 0.67 0.33

Exploitability E is calculated using the following formula:

$$E = \prod m_e$$

4.2 IMPACT

CONFIDENTIALITY (C):

Measures the impact to the confidentiality of the information resources managed by the contract due to a successfully exploited vulnerability. Confidentiality refers to limiting access to authorized users only.

INTEGRITY (I):

Measures the impact to integrity of a successfully exploited vulnerability. Integrity refers to the trustworthiness and veracity of data stored and/or processed on-chain. Integrity impact directly affecting Deposit or Yield records is excluded.

AVAILABILITY (A):

Measures the impact to the availability of the impacted component resulting from a successfully exploited vulnerability. This metric refers to smart contract features and functionality, not state. Availability impact directly affecting Deposit or Yield is excluded.

DEPOSIT (D):

Measures the impact to the deposits made to the contract by either users or owners.

YIELD (Y):

Measures the impact to the yield generated by the contract for either users or owners.

METRICS:

IMPACT METRIC (M_I)	METRIC VALUE	NUMERICAL VALUE
Confidentiality (C)	None (I:N) Low (I:L) Medium (I:M) High (I:H) Critical (I:C)	0 0.25 0.5 0.75 1
Integrity (I)	None (I:N) Low (I:L) Medium (I:M) High (I:H) Critical (I:C)	0 0.25 0.5 0.75 1
Availability (A)	None (A:N) Low (A:L) Medium (A:M) High (A:H) Critical (A:C)	0 0.25 0.5 0.75 1
Deposit (D)	None (D:N) Low (D:L) Medium (D:M) High (D:H) Critical (D:C)	0 0.25 0.5 0.75 1
Yield (Y)	None (Y:N) Low (Y:L) Medium (Y:M) High (Y:H) Critical (Y:C)	0 0.25 0.5 0.75 1

Impact I is calculated using the following formula:

$$I = max(m_I) + \frac{\sum m_I - max(m_I)}{4}$$

4.3 SEVERITY COEFFICIENT

REVERSIBILITY (R):

Describes the share of the exploited vulnerability effects that can be reversed. For upgradeable contracts, assume the contract private key is available.

SCOPE (S):

Captures whether a vulnerability in one vulnerable contract impacts resources in other contracts.

METRICS:

SEVERITY COEFFICIENT (C)	COEFFICIENT VALUE	NUMERICAL VALUE
Reversibility (r)	None (R:N) Partial (R:P) Full (R:F)	1 0.5 0.25

SEVERITY COEFFICIENT (C)	COEFFICIENT VALUE	NUMERICAL VALUE
Scope (s)	Changed (S:C) Unchanged (S:U)	1.25 1

Severity Coefficient C is obtained by the following product:

$$C = rs$$

The Vulnerability Severity Score S is obtained by:

$$S = min(10, EIC * 10)$$

The score is rounded up to 1 decimal places.

SEVERITY	SCORE VALUE RANGE
Critical	9 - 10
High	7 - 8.9
Medium	4.5 - 6.9
Low	2 - 4.4
Informational	0 - 1.9

5. SCOPE

FILES AND REPOSITORY

(a) Repository: glow-v1

(b) Assessed Commit ID: 47367ed

(c) Items in scope:

- ./programs/margin-pool/Cargo.toml
- ./programs/margin-pool/Xargo.toml
- ./programs/margin-pool/src/instructions/margin_refresh_position.rs
- ./programs/margin-pool/src/instructions/close_loan.rs
- ./programs/margin-pool/src/instructions/configure.rs
- ./programs/margin-pool/src/instructions/withdraw.rs
- ./programs/margin-pool/src/instructions/repay.rs
- ./programs/margin-pool/src/instructions/margin_borrow_v2.rs
- ./programs/margin-pool/src/instructions/deposit.rs
- ./programs/margin-pool/src/instructions/admin/mod.rs
- ./programs/margin-pool/src/instructions/admin/admin_transfer_loan.rs
- ./programs/margin-pool/src/instructions/collect.rs
- ./programs/margin-pool/src/instructions/margin_repay.rs
- ./programs/margin-pool/src/instructions/create_pool.rs
- ./programs/margin-pool/src/instructions/margin_borrow.rs
- ./programs/margin-pool/src/instructions/register_loan.rs
- ./programs/margin-pool/src/instructions.rs
- ./programs/margin-pool/src/events.rs
- ./programs/margin-pool/src/util.rs
- ./programs/margin-pool/src/lib.rs
- ./programs/margin-pool/src/state.rs
- ./programs/airspace/Cargo.toml
- ./programs/airspace/Xargo.toml
- ./programs/airspace/src/instructions/airspace_permit_revoke.rs
- ./programs/airspace/src/instructions/airspace_permit_issuer_revoke.rs
- ./programs/airspace/src/instructions/airspace_set_authority.rs
- ./programs/airspace/src/instructions/airspace_create.rs
- ./programs/airspace/src/instructions/create_governor_id.rs
- ./programs/airspace/src/instructions/set_governor.rs
- ./programs/airspace/src/instructions/mod.rs
- ./programs/airspace/src/instructions/airspace_permit_issuer_create.rs
- ./programs/airspace/src/instructions/airspace_permit_create.rs
- ./programs/airspace/src/events.rs
- ./programs/airspace/src/lib.rs
- ./programs/airspace/src/state.rs
- ./programs/margin/Cargo.toml

- ./programs/margin/Xargo.toml
- ./programs/margin/src/instructions/accounting_invoke.rs
- ./programs/margin/src/instructions/configure/configure_account_airspace.rs
- ./programs/margin/src/instructions/configure/configure_adapter.rs
- ./programs/margin/src/instructions/configure/mod.rs
- ./programs/margin/src/instructions/configure/configure_permit.rs
- ./programs/margin/src/instructions/configure/configure_token.rs
- ./programs/margin/src/instructions/register_position.rs
- ./programs/margin/src/instructions/verify_healthy.rs
- ./programs/margin/src/instructions/admin/admin_transfer_position.rs
- ./programs/margin/src/instructions/admin/mod.rs
- ./programs/margin/src/instructions/adapter_invoke.rs
- ./programs/margin/src/instructions/verify_unhealthy.rs
- ./programs/margin/src/instructions/close_position.rs
- ./programs/margin/src/instructions/liquidate_end.rs
- ./programs/margin/src/instructions/create_account.rs
- ./programs/margin/src/instructions/update_position_balance.rs
- ./programs/margin/src/instructions/liquidator_invoke.rs
- ./programs/margin/src/instructions/liquidate_begin.rs
- ./programs/margin/src/instructions/positions/refresh_deposit_position.rs
- ./programs/margin/src/instructions/positions/transfer_deposit.rs
- ./programs/margin/src/instructions/positions/create_deposit_position.rs
- ./programs/margin/src/instructions/positions/mod.rs
- ./programs/margin/src/instructions/positions/refresh_position_config.rs
- ./programs/margin/src/instructions/lookup_tables/create_lookup_table.rs
- ./programs/margin/src/instructions/lookup_tables/append_to_lookup.rs
- ./programs/margin/src/instructions/lookup_tables/mod.rs
- ./programs/margin/src/instructions/lookup_tables/init_lookup_registry.rs
- ./programs/margin/src/instructions/close_account.rs
- ./programs/margin/src/instructions.rs
- ./programs/margin/src/adapter.rs
- ./programs/margin/src/events.rs
- ./programs/margin/src/util.rs
- ./programs/margin/src/lib.rs
- ./programs/margin/src/state/config.rs
- ./programs/margin/src/state/account.rs
- ./programs/margin/src/state/account/positions.rs
- ./programs/margin/src/state.rs
- ./programs/margin/src/seeds.rs
- ./programs/margin/src/syscall.rs
- ./programs/metadata/Cargo.toml
- ./programs/metadata/Xargo.toml
- ./programs/metadata/src/lib.rs
- ./programs/control/Cargo.toml
- ./programs/control/Xargo.toml
- ./programs/control/src/instructions/create_margin_pool.rs

- ./programs/control/src/instructions/create_authority.rs
- ./programs/control/src/instructions/configure_margin_pool.rs
- ./programs/control/src/instructions.rs
- ./programs/control/src/events.rs
- ./programs/control/src/lib.rs
- ./Cargo.toml
- ./Anchor.toml
- ./libraries/rust/instructions/src/airspace.rs
- ./libraries/rust/program-proc-macros/Cargo.toml
- ./libraries/rust/program-proc-macros/src/lib.rs
- ./libraries/rust/program-proc-macros/src/mem.rs
- ./libraries/rust/instructions/Cargo.toml
- ./libraries/rust/instructions/src/margin.rs
- ./libraries/rust/instructions/src/control.rs
- ./libraries/rust/instructions/src/lib.rs
- ./libraries/rust/instructions/src/test_service.rs
- ./libraries/rust/instructions/src/margin_pool.rs
- ./libraries/rust/tools/Cargo.toml
- ./libraries/rust/tools/src/lookup_tables.rs
- ./libraries/rust/tools/src/lib.rs
- ./libraries/rust/solana-client/Cargo.toml
- ./libraries/rust/program-common/Cargo.toml
- ./libraries/rust/program-common/src/pod.rs
- ./libraries/rust/program-common/src/number_128.rs
- ./libraries/rust/program-common/src/fp32.rs
- ./libraries/rust/program-common/src/number.rs
- ./libraries/rust/program-common/src/log.rs
- ./libraries/rust/program-common/src/lib.rs
- ./libraries/rust/program-common/src/functions.rs
- ./libraries/rust/program-common/src/serialization.rs
- ./libraries/rust/program-common/src/interest_pricing.rs
- ./libraries/rust/program-common/src/traits.rs
- ./libraries/rust/program-common/src/valuation.rs

Out-of-Scope: ./programs/test-service/Cargo.toml, ./programs/test-service/Xargo.toml, ./programs/test-service/src/instructions/slippy/mod.rs, ./programs/test-service/src/instructions/mod.rs, ./programs/test-service/src/instructions/if_not_initialized.rs, ./programs/test-service/src/instructions/tokens/token_update_pyth_price.rs, ./programs/test-service/src/instructions/tokens/token_register.rs, ./programs/test-service/src/instructions/tokens/token_create.rs, ./programs/test-service/src/instructions/tokens/mod.rs, ./programs/test-service/src/instructions/tokens/token_init_native.rs, ./programs/test-service/src/instructions/tokens/token_request.rs, ./programs/test-service/src/util.rs, ./programs/test-service/src/error.rs, ./programs/test-service/src/lib.rs, ./programs/test-

service/src/state/tokens.rs, ./programs/test-service/src/state/mod.rs, ./programs/test-service/src/state/slippy.rs, ./programs/turbo.json, ./programs/package.json, ./libraries/rust/tools/Cargo.toml, ./libraries/rust/tools/src/lookup_tables.rs, ./libraries/rust/tools/src/lib.rs, ./libraries/rust/solana-client/Cargo.toml, ./libraries/rust/solana-client/src/lookup_tables.rs, ./libraries/rust/solana-client/src/transaction.rs, ./libraries/rust/solana-client/src/util/pubkey.rs, ./libraries/rust/solana-client/src/util/mod.rs, ./libraries/rust/solana-client/src/util/data.rs, ./libraries/rust/solana-client/src/util/keypair.rs, ./libraries/rust/solana-client/src/lib.rs, ./libraries/rust/solana-client/src/signature.rs, ./libraries/rust/solana-client/src/rpc/native.rs, ./libraries/rust/solana-client/src/network.rs, ./libraries/rust/solana-client/src/rpc.rs, ./libraries/rust/client-web/Cargo.toml, ./libraries/rust/client-web/src/margin.rs, ./libraries/rust/client-web/src/solana_web3.rs, ./libraries/rust/client-web/src/error.rs, ./libraries/rust/client-web/src/lib.rs, ./libraries/rust/client-web/src/margin_pool.rs, ./libraries/rust/client-web/src/wallet.rs, ./libraries/rust/simulation/Cargo.toml, ./libraries/rust/simulation/src/runtime.rs, ./libraries/rust/simulation/src/lib.rs, ./libraries/rust/simulation/src/solana_rpc_api.rs, ./libraries/rust/testing/Cargo.toml, ./libraries/rust/testing/src/lib.rs, ./libraries/rust/environment/Cargo.toml, ./libraries/rust/environment/src/lookup_tables.rs, ./libraries/rust/environment/src/config.rs, ./libraries/rust/environment/src/lib.rs, ./libraries/rust/environment/src/builder/margin.rs, ./libraries/rust/environment/src/builder/global.rs, ./libraries/rust/environment/src/builder/margin_pool.rs, ./libraries/rust/environment/src/client_config.rs, ./libraries/rust/environment/src/builder.rs, ./libraries/rust/margin/Cargo.toml, ./libraries/rust/margin/src/margin_account_ext.rs, ./libraries/rust/margin/src/ix_builder.rs, ./libraries/rust/margin/src/lookup_tables.rs, ./libraries/rust/margin/src/tokens.rs, ./libraries/rust/margin/src/util/queue_processor.rs, ./libraries/rust/margin/src/util/asynchronous.rs, ./libraries/rust/margin/src/util/no_dupe_queue.rs, ./libraries/rust/margin/src/util/mod.rs, ./libraries/rust/margin/src/refresh/position_refresher.rs, ./libraries/rust/margin/src/refresh/deposit.rs, ./libraries/rust/margin/src/refresh/mod.rs, ./libraries/rust/margin/src/refresh/pool.rs, ./libraries/rust/margin/src/lib.rs, ./libraries/rust/margin/src/test_service.rs, ./libraries/rust/margin/src/solana/transaction.rs, ./libraries/rust/margin/src/solana/mod.rs, ./libraries/rust/margin/src/tx_builder/airspace.rs, ./libraries/rust/margin/src/tx_builder/invoke_pool.rs, ./libraries/rust/margin/src/tx_builder/user.rs, ./libraries/rust/margin/src/tx_builder/invoke_context.rs, ./libraries/rust/margin/src/tx_builder/mod.rs, ./libraries/rust/margin/src/get_state/rpc_query.rs, ./libraries/rust/margin/src/get_state/margin_state.rs, ./libraries/rust/margin/src/get_state/mod.rs, ./libraries/rust/margin/src/margin_integrator.rs, ./libraries/rust/static-program-registry/Cargo.toml, ./libraries/rust/static-program-registry/src/lib.rs, ./libraries/rust/client/Cargo.toml, ./libraries/rust/client/src/margin.rs, ./libraries/rust/client/src/client.rs, ./libraries/rust/client/src/config.rs, ./libraries/rust/client/src/lib.rs, ./libraries/rust/client/src/test_service.rs,

./libraries/rust/client/src/margin_pool.rs, ./libraries/rust/client/src/state/lookup_tables.rs, ./libraries/rust/client/src/state/margin.rs, ./libraries/rust/client/src/state/tokens.rs, ./libraries/rust/client/src/state/oracles.rs, ./libraries/rust/client/src/state/margin_pool.rs, ./libraries/rust/client/src/wallet.rs, ./libraries/rust/client/src/state.rs, ./libraries/rust/client/src/fixed_term/util.rs, third party dependencies and economic attacks.

FILES AND REPOSITORY

(a) Repository: lookup-table-registry

(b) Assessed Commit ID: 923f5a3

(c) Items in scope:

- ./programs/lookup-table-registry/Cargo.toml
- ./programs/lookup-table-registry/Xargo.toml
- ./programs/lookup-table-registry/src/lib.rs
- ./programs/lookup-table-registry/src/state.rs
- ./Cargo.toml
- ./Anchor.toml

Out-of-Scope: ./rust-toolchain, ./LICENSE, ./Dockerfile, ./tests/lookup-table-registry.ts, ./Cargo.lock, ./libraries/rust/Cargo.toml, ./libraries/rust/src/instructions.rs, ./libraries/rust/src/lib.rs, ./libraries/rust/src/writer.rs, ./libraries/rust/src/reader.rs, ./libraries/rust/src/common.rs, ./libraries/solana-address-lookup-table-program-gateway/Cargo.toml, ./libraries/solana-address-lookup-table-program-gateway/src/lib.rs, ./libraries/solana-address-lookup-table-program-gateway/src/stub/state.rs, ./libraries/solana-address-lookup-table-program-gateway/src/stub/instruction.rs, ./server/Cargo.toml, ./server/src/main.rs, ./.prettierignore, ./README.md, ./test-validator, ./.env.template, ./yarn.lock, ./.dockerignore, ./.gitignore, ./package.json, ./.github/workflows/build.yaml, ./tsconfig.json, third party dependencies and economic attacks.

FILES AND REPOSITORY

(a) Repository: glow-v1

(b) Assessed Commit ID: https://github.com/Blueprint-Finance/glow-v1/pull/1196

(c) Items in scope:

- ./programs/margin-pool/Cargo.toml
- ./programs/margin-pool/Xargo.toml
- ./programs/margin-pool/src/instructions/margin_refresh_position.rs
- ./programs/margin-pool/src/instructions/close_loan.rs

- ./programs/margin-pool/src/instructions/configure.rs
- ./programs/margin-pool/src/instructions/withdraw.rs
- ./programs/margin-pool/src/instructions/repay.rs
- ./programs/margin-pool/src/instructions/margin_borrow_v2.rs
- ./programs/margin-pool/src/instructions/deposit.rs
- ./programs/margin-pool/src/instructions/admin/mod.rs
- ./programs/margin-pool/src/instructions/admin/admin_transfer_loan.rs
- ./programs/margin-pool/src/instructions/collect.rs
- ./programs/margin-pool/src/instructions/margin_repay.rs
- ./programs/margin-pool/src/instructions/create_pool.rs
- ./programs/margin-pool/src/instructions/margin_borrow.rs
- ./programs/margin-pool/src/instructions/register_loan.rs
- ./programs/margin-pool/src/instructions/withdraw_fees.rs
- ./programs/margin-pool/src/instructions.rs
- ./programs/margin-pool/src/events.rs
- ./programs/margin-pool/src/util.rs
- ./programs/margin-pool/src/lib.rs
- ./programs/margin-pool/src/state.rs
- ./programs/airspace/Cargo.toml
- ./programs/airspace/Xargo.toml
- ./programs/airspace/src/instructions/airspace_permit_revoke.rs
- ./programs/airspace/src/instructions/airspace_permit_issuer_revoke.rs
- ./programs/airspace/src/instructions/airspace_set_authority.rs
- ./programs/airspace/src/instructions/airspace_create.rs
- ./programs/airspace/src/instructions/create_governor_id.rs
- ./programs/airspace/src/instructions/set_governor.rs
- ./programs/airspace/src/instructions/mod.rs
- ./programs/airspace/src/instructions/airspace_permit_issuer_create.rs
- ./programs/airspace/src/instructions/airspace_permit_create.rs
- ./programs/airspace/src/events.rs
- ./programs/airspace/src/lib.rs
- ./programs/airspace/src/state.rs
- ./programs/margin/Cargo.toml
- ./programs/margin/Xargo.toml
- ./programs/margin/src/instructions/accounting_invoke.rs
- ./programs/margin/src/instructions/configure/configure_account_airspace.rs
- ./programs/margin/src/instructions/configure/configure_adapter.rs
- ./programs/margin/src/instructions/configure/mod.rs
- ./programs/margin/src/instructions/configure/configure_permit.rs
- ./programs/margin/src/instructions/configure/configure_token.rs
- ./programs/margin/src/instructions/register_position.rs
- ./programs/margin/src/instructions/verify_healthy.rs
- ./programs/margin/src/instructions/admin/admin_transfer_position.rs
- ./programs/margin/src/instructions/admin/mod.rs
- ./programs/margin/src/instructions/adapter_invoke.rs
- ./programs/margin/src/instructions/verify_unhealthy.rs

- ./programs/margin/src/instructions/close_position.rs
- ./programs/margin/src/instructions/liquidate_end.rs
- ./programs/margin/src/instructions/create_account.rs
- ./programs/margin/src/instructions/update_position_balance.rs
- ./programs/margin/src/instructions/liquidator_invoke.rs
- ./programs/margin/src/instructions/liquidate_begin.rs
- ./programs/margin/src/instructions/positions/refresh_deposit_position.rs
- ./programs/margin/src/instructions/positions/transfer_deposit.rs
- ./programs/margin/src/instructions/positions/create_deposit_position.rs
- ./programs/margin/src/instructions/positions/mod.rs
- ./programs/margin/src/instructions/positions/refresh_position_config.rs
- ./programs/margin/src/instructions/lookup_tables/create_lookup_table.rs
- ./programs/margin/src/instructions/lookup_tables/append_to_lookup.rs
- ./programs/margin/src/instructions/lookup_tables/mod.rs
- ./programs/margin/src/instructions/lookup_tables/init_lookup_registry.rs
- ./programs/margin/src/instructions/close_account.rs
- ./programs/margin/src/instructions.rs
- ./programs/margin/src/adapter.rs
- ./programs/margin/src/events.rs
- ./programs/margin/src/util.rs
- ./programs/margin/src/lib.rs
- ./programs/margin/src/state/config.rs
- ./programs/margin/src/state/account.rs
- ./programs/margin/src/state/account/positions.rs
- ./programs/margin/src/state.rs
- ./programs/margin/src/seeds.rs
- ./programs/margin/src/syscall.rs
- ./programs/metadata/Cargo.toml
- ./programs/metadata/Xargo.toml
- ./programs/metadata/src/lib.rs
- ./Cargo.toml
- ./Anchor.toml
- ./libraries/rust/instructions/src/airspace.rs
- ./libraries/rust/program-proc-macros/Cargo.toml
- ./libraries/rust/program-proc-macros/src/lib.rs
- ./libraries/rust/program-proc-macros/src/mem.rs
- ./libraries/rust/instructions/Cargo.toml
- ./libraries/rust/instructions/src/margin.rs
- ./libraries/rust/instructions/src/control.rs
- ./libraries/rust/instructions/src/lib.rs
- ./libraries/rust/instructions/src/test_service.rs
- ./libraries/rust/instructions/src/margin_pool.rs
- ./libraries/rust/tools/Cargo.toml
- ./libraries/rust/tools/src/lookup_tables.rs
- ./libraries/rust/tools/src/lib.rs
- ./libraries/rust/solana-client/Cargo.toml

- ./libraries/rust/program-common/Cargo.toml
- ./libraries/rust/program-common/src/pod.rs
- ./libraries/rust/program-common/src/number_128.rs
- ./libraries/rust/program-common/src/fp32.rs
- ./libraries/rust/program-common/src/number.rs
- ./libraries/rust/program-common/src/log.rs
- ./libraries/rust/program-common/src/lib.rs
- ./libraries/rust/program-common/src/functions.rs
- ./libraries/rust/program-common/src/serialization.rs
- ./libraries/rust/program-common/src/interest_pricing.rs
- ./libraries/rust/program-common/src/traits.rs
- ./libraries/rust/program-common/src/valuation.rs

Out-of-Scope: ./programs/test-service/Cargo.toml, ./programs/test-service/Xargo.toml, ./programs/test-service/src/instructions/slippy/mod.rs, ./programs/test-service/src/instructions/mod.rs, ./programs/test-service/src/instructions/if_not_initialized.rs, ./programs/test-service/src/instructions/tokens/token_update_pyth_price.rs, ./programs/test-service/src/instructions/tokens/token_register.rs, ./programs/test-service/src/instructions/tokens/token_create.rs, ./programs/test-service/src/instructions/tokens/mod.rs, ./programs/test-service/src/instructions/tokens/token_init_native.rs, ./programs/test-service/src/instructions/tokens/token_request.rs, ./programs/test-service/src/util.rs, ./programs/test-service/src/error.rs, ./programs/test-service/src/lib.rs, ./programs/test-service/src/state/tokens.rs, ./programs/test-service/src/state/mod.rs, ./programs/test-service/src/state/slippy.rs, ./programs/turbo.json, ./programs/package.json, ./libraries/rust/tools/Cargo.toml, ./libraries/rust/tools/src/lookup_tables.rs, ./libraries/rust/tools/src/lib.rs, ./libraries/rust/solana-client/Cargo.toml, ./libraries/rust/solana-client/src/lookup_tables.rs, ./libraries/rust/solana-client/src/transaction.rs, ./libraries/rust/solana-client/src/util/pubkey.rs, ./libraries/rust/solana-client/src/util/mod.rs, ./libraries/rust/solana-client/src/util/data.rs, ./libraries/rust/solana-client/src/util/keypair.rs, ./libraries/rust/solana-client/src/lib.rs, ./libraries/rust/solana-client/src/signature.rs, ./libraries/rust/solana-client/src/rpc/native.rs, ./libraries/rust/solana-client/src/network.rs, ./libraries/rust/solana-client/src/rpc.rs, ./libraries/rust/client-web/Cargo.toml, ./libraries/rust/client-web/src/margin.rs, ./libraries/rust/client-web/src/solana_web3.rs, ./libraries/rust/client-web/src/error.rs, ./libraries/rust/client-web/src/lib.rs, ./libraries/rust/client-web/src/margin_pool.rs, ./libraries/rust/client-web/src/wallet.rs, ./libraries/rust/simulation/Cargo.toml, ./libraries/rust/simulation/src/runtime.rs, ./libraries/rust/simulation/src/lib.rs, ./libraries/rust/simulation/src/solana_rpc_api.rs, ./libraries/rust/testing/Cargo.toml, ./libraries/rust/testing/src/lib.rs, ./libraries/rust/environment/Cargo.toml, ./libraries/rust/environment/src/lookup_tables.rs, ./libraries/rust/environment/src/config.rs, ./libraries/rust/environment/src/lib.rs, ./libraries/rust/environment/src/builder/margin.rs, ./libraries/rust/environment/src/builder/global.rs,

./libraries/rust/environment/src/builder/margin_pool.rs, ./libraries/rust/environment/src/client_config.rs, ./libraries/rust/environment/src/builder.rs, ./libraries/rust/margin/Cargo.toml, ./libraries/rust/margin/src/margin_account_ext.rs, ./libraries/rust/margin/src/ix_builder.rs, ./libraries/rust/margin/src/lookup_tables.rs, ./libraries/rust/margin/src/tokens.rs, ./libraries/rust/margin/src/util/queue_processor.rs, ./libraries/rust/margin/src/util/asynchronous.rs, ./libraries/rust/margin/src/util/no_dupe_queue.rs, ./libraries/rust/margin/src/util/mod.rs, ./libraries/rust/margin/src/refresh/position_refresher.rs, ./libraries/rust/margin/src/refresh/deposit.rs, ./libraries/rust/margin/src/refresh/mod.rs, ./libraries/rust/margin/src/refresh/pool.rs, ./libraries/rust/margin/src/lib.rs, ./libraries/rust/margin/src/test_service.rs, ./libraries/rust/margin/src/solana/transaction.rs, ./libraries/rust/margin/src/solana/mod.rs, ./libraries/rust/margin/src/tx_builder/airspace.rs, ./libraries/rust/margin/src/tx_builder/invoke_pool.rs, ./libraries/rust/margin/src/tx_builder/user.rs, ./libraries/rust/margin/src/tx_builder/invoke_context.rs, ./libraries/rust/margin/src/tx_builder/mod.rs, ./libraries/rust/margin/src/get_state/rpc_query.rs, ./libraries/rust/margin/src/get_state/margin_state.rs, ./libraries/rust/margin/src/get_state/mod.rs, ./libraries/rust/margin/src/margin_integrator.rs, ./libraries/rust/static-program-registry/Cargo.toml, ./libraries/rust/static-program-registry/src/lib.rs, ./libraries/rust/client/Cargo.toml, ./libraries/rust/client/src/margin.rs, ./libraries/rust/client/src/client.rs, ./libraries/rust/client/src/config.rs, ./libraries/rust/client/src/lib.rs, ./libraries/rust/client/src/test_service.rs, ./libraries/rust/client/src/margin_pool.rs, ./libraries/rust/client/src/state/lookup_tables.rs, ./libraries/rust/client/src/state/margin.rs, ./libraries/rust/client/src/state/tokens.rs, ./libraries/rust/client/src/state/oracles.rs, ./libraries/rust/client/src/state/margin_pool.rs, ./libraries/rust/client/src/wallet.rs, ./libraries/rust/client/src/state.rs, ./libraries/rust/client/src/fixed_term/util.rs, third party dependencies and economic attacks.

FILES AND REPOSITORY

(a) Repository: glow-v1

(b) Assessed Commit ID: https://github.com/Blueprint-Finance/glow-v1/pull/1218

(c) Items in scope:

- ./programs/margin-pool/Cargo.toml
- ./programs/margin-pool/Xargo.toml
- ./programs/margin-pool/src/instructions/margin_refresh_position.rs
- ./programs/margin-pool/src/instructions/close_loan.rs
- ./programs/margin-pool/src/instructions/configure.rs
- ./programs/margin-pool/src/instructions/withdraw.rs
- ./programs/margin-pool/src/instructions/repay.rs

- ./programs/margin-pool/src/instructions/margin_borrow_v2.rs
- ./programs/margin-pool/src/instructions/deposit.rs
- ./programs/margin-pool/src/instructions/admin/mod.rs
- ./programs/margin-pool/src/instructions/admin/admin_transfer_loan.rs
- ./programs/margin-pool/src/instructions/collect.rs
- ./programs/margin-pool/src/instructions/margin_repay.rs
- ./programs/margin-pool/src/instructions/create_pool.rs
- ./programs/margin-pool/src/instructions/margin_borrow.rs
- ./programs/margin-pool/src/instructions/register_loan.rs
- ./programs/margin-pool/src/instructions/withdraw_fees.rs
- ./programs/margin-pool/src/instructions.rs
- ./programs/margin-pool/src/events.rs
- ./programs/margin-pool/src/util.rs
- ./programs/margin-pool/src/lib.rs
- ./programs/margin-pool/src/state.rs
- ./programs/airspace/Cargo.toml
- ./programs/airspace/Xargo.toml
- ./programs/airspace/src/instructions/airspace_permit_revoke.rs
- ./programs/airspace/src/instructions/airspace_permit_issuer_revoke.rs
- ./programs/airspace/src/instructions/airspace_set_authority.rs
- ./programs/airspace/src/instructions/airspace_create.rs
- ./programs/airspace/src/instructions/create_governor_id.rs
- ./programs/airspace/src/instructions/set_governor.rs
- ./programs/airspace/src/instructions/mod.rs
- ./programs/airspace/src/instructions/airspace_permit_issuer_create.rs
- ./programs/airspace/src/instructions/airspace_permit_create.rs
- ./programs/airspace/src/events.rs
- ./programs/airspace/src/lib.rs
- ./programs/airspace/src/state.rs
- ./programs/margin/Cargo.toml
- ./programs/margin/Xargo.toml
- ./programs/margin/src/instructions/accounting_invoke.rs
- ./programs/margin/src/instructions/configure/configure_account_airspace.rs
- ./programs/margin/src/instructions/configure/configure_adapter.rs
- ./programs/margin/src/instructions/configure/mod.rs
- ./programs/margin/src/instructions/configure/configure_permit.rs
- ./programs/margin/src/instructions/configure/configure_token.rs
- ./programs/margin/src/instructions/register_position.rs
- ./programs/margin/src/instructions/verify_healthy.rs
- ./programs/margin/src/instructions/admin/admin_transfer_position.rs
- ./programs/margin/src/instructions/admin/mod.rs
- ./programs/margin/src/instructions/adapter_invoke.rs
- ./programs/margin/src/instructions/verify_unhealthy.rs
- ./programs/margin/src/instructions/close_position.rs
- ./programs/margin/src/instructions/liquidate_end.rs
- ./programs/margin/src/instructions/create_account.rs

- ./programs/margin/src/instructions/update_position_balance.rs
- ./programs/margin/src/instructions/liquidator_invoke.rs
- ./programs/margin/src/instructions/liquidate_begin.rs
- ./programs/margin/src/instructions/positions/refresh_deposit_position.rs
- ./programs/margin/src/instructions/positions/transfer_deposit.rs
- ./programs/margin/src/instructions/positions/create_deposit_position.rs
- ./programs/margin/src/instructions/positions/mod.rs
- ./programs/margin/src/instructions/positions/refresh_position_config.rs
- ./programs/margin/src/instructions/lookup_tables/create_lookup_table.rs
- ./programs/margin/src/instructions/lookup_tables/append_to_lookup.rs
- ./programs/margin/src/instructions/lookup_tables/mod.rs
- ./programs/margin/src/instructions/lookup_tables/init_lookup_registry.rs
- ./programs/margin/src/instructions/close_account.rs
- ./programs/margin/src/instructions.rs
- ./programs/margin/src/adapter.rs
- ./programs/margin/src/events.rs
- ./programs/margin/src/util.rs
- ./programs/margin/src/lib.rs
- ./programs/margin/src/state/config.rs
- ./programs/margin/src/state/account.rs
- ./programs/margin/src/state/account/positions.rs
- ./programs/margin/src/state.rs
- ./programs/margin/src/seeds.rs
- ./programs/margin/src/syscall.rs
- ./programs/metadata/Cargo.toml
- ./programs/metadata/Xargo.toml
- ./programs/metadata/src/lib.rs
- ./Cargo.toml
- ./Anchor.toml
- ./libraries/rust/instructions/src/airspace.rs
- ./libraries/rust/program-proc-macros/Cargo.toml
- ./libraries/rust/program-proc-macros/src/lib.rs
- ./libraries/rust/program-proc-macros/src/mem.rs
- ./libraries/rust/instructions/Cargo.toml
- ./libraries/rust/instructions/src/margin.rs
- ./libraries/rust/instructions/src/control.rs
- ./libraries/rust/instructions/src/lib.rs
- ./libraries/rust/instructions/src/test_service.rs
- ./libraries/rust/instructions/src/margin_pool.rs
- ./libraries/rust/tools/Cargo.toml
- ./libraries/rust/tools/src/lookup_tables.rs
- ./libraries/rust/tools/src/lib.rs
- ./libraries/rust/solana-client/Cargo.toml
- ./libraries/rust/program-common/Cargo.toml
- ./libraries/rust/program-common/src/pod.rs
- ./libraries/rust/program-common/src/number_128.rs

- ./libraries/rust/program-common/src/fp32.rs
- ./libraries/rust/program-common/src/number.rs
- ./libraries/rust/program-common/src/log.rs
- ./libraries/rust/program-common/src/lib.rs
- ./libraries/rust/program-common/src/functions.rs
- ./libraries/rust/program-common/src/serialization.rs
- ./libraries/rust/program-common/src/interest_pricing.rs
- ./libraries/rust/program-common/src/traits.rs
- ./libraries/rust/program-common/src/valuation.rs
- libraries/rust/program-common/src/oracle.rs

Out-of-Scope: ./programs/test-service/Cargo.toml, ./programs/test-service/Xargo.toml, ./programs/test-service/src/instructions/slippy/mod.rs, ./programs/test-service/src/instructions/mod.rs, ./programs/test-service/src/instructions/if_not_initialized.rs, ./programs/test-service/src/instructions/tokens/token_update_pyth_price.rs, ./programs/test-service/src/instructions/tokens/token_register.rs, ./programs/test-service/src/instructions/tokens/token_create.rs, ./programs/test-service/src/instructions/tokens/mod.rs, ./programs/test-service/src/instructions/tokens/token_init_native.rs, ./programs/test-service/src/instructions/tokens/token_request.rs, ./programs/test-service/src/util.rs, ./programs/test-service/src/error.rs, ./programs/test-service/src/lib.rs, ./programs/test-service/src/state/tokens.rs, ./programs/test-service/src/state/mod.rs, ./programs/test-service/src/state/slippy.rs, ./programs/turbo.json, ./programs/package.json, ./libraries/rust/tools/Cargo.toml, ./libraries/rust/tools/src/lookup_tables.rs, ./libraries/rust/tools/src/lib.rs, ./libraries/rust/solana-client/Cargo.toml, ./libraries/rust/solana-client/src/lookup_tables.rs, ./libraries/rust/solana-client/src/transaction.rs, ./libraries/rust/solana-client/src/util/pubkey.rs, ./libraries/rust/solana-client/src/util/mod.rs, ./libraries/rust/solana-client/src/util/data.rs, ./libraries/rust/solana-client/src/util/keypair.rs, ./libraries/rust/solana-client/src/lib.rs, ./libraries/rust/solana-client/src/signature.rs, ./libraries/rust/solana-client/src/rpc/native.rs, ./libraries/rust/solana-client/src/network.rs, ./libraries/rust/solana-client/src/rpc.rs, ./libraries/rust/client-web/Cargo.toml, ./libraries/rust/client-web/src/margin.rs, ./libraries/rust/client-web/src/solana_web3.rs, ./libraries/rust/client-web/src/error.rs, ./libraries/rust/client-web/src/lib.rs, ./libraries/rust/client-web/src/margin_pool.rs, ./libraries/rust/client-web/src/wallet.rs, ./libraries/rust/simulation/Cargo.toml, ./libraries/rust/simulation/src/runtime.rs, ./libraries/rust/simulation/src/lib.rs, ./libraries/rust/simulation/src/solana_rpc_api.rs, ./libraries/rust/testing/Cargo.toml, ./libraries/rust/testing/src/lib.rs, ./libraries/rust/environment/Cargo.toml, ./libraries/rust/environment/src/lookup_tables.rs, ./libraries/rust/environment/src/config.rs, ./libraries/rust/environment/src/lib.rs, ./libraries/rust/environment/src/builder/margin.rs, ./libraries/rust/environment/src/builder/global.rs, ./libraries/rust/environment/src/builder/margin_pool.rs, ./libraries/rust/environment/src/client_config.rs, ./libraries/rust/environment/src/builder.rs,

./libraries/rust/margin/Cargo.toml, ./libraries/rust/margin/src/margin_account_ext.rs, ./libraries/rust/margin/src/ix_builder.rs, ./libraries/rust/margin/src/lookup_tables.rs, ./libraries/rust/margin/src/tokens.rs, ./libraries/rust/margin/src/util/queue_processor.rs, ./libraries/rust/margin/src/util/asynchronous.rs, ./libraries/rust/margin/src/util/no_dupe_queue.rs, ./libraries/rust/margin/src/util/mod.rs, ./libraries/rust/margin/src/refresh/position_refresher.rs, ./libraries/rust/margin/src/refresh/deposit.rs, ./libraries/rust/margin/src/refresh/mod.rs, ./libraries/rust/margin/src/refresh/pool.rs, ./libraries/rust/margin/src/lib.rs, ./libraries/rust/margin/src/test_service.rs, ./libraries/rust/margin/src/solana/transaction.rs, ./libraries/rust/margin/src/solana/mod.rs, ./libraries/rust/margin/src/tx_builder/airspace.rs, ./libraries/rust/margin/src/tx_builder/invoke_pool.rs, ./libraries/rust/margin/src/tx_builder/user.rs, ./libraries/rust/margin/src/tx_builder/invoke_context.rs, ./libraries/rust/margin/src/tx_builder/mod.rs, ./libraries/rust/margin/src/get_state/rpc_query.rs, ./libraries/rust/margin/src/get_state/margin_state.rs, ./libraries/rust/margin/src/get_state/mod.rs, ./libraries/rust/margin/src/margin_integrator.rs, ./libraries/rust/static-program-registry/Cargo.toml, ./libraries/rust/static-program-registry/src/lib.rs, ./libraries/rust/client/Cargo.toml, ./libraries/rust/client/src/margin.rs, ./libraries/rust/client/src/client.rs, ./libraries/rust/client/src/config.rs, ./libraries/rust/client/src/lib.rs, ./libraries/rust/client/src/test_service.rs, ./libraries/rust/client/src/margin_pool.rs, ./libraries/rust/client/src/state/lookup_tables.rs, ./libraries/rust/client/src/state/margin.rs, ./libraries/rust/client/src/state/tokens.rs, ./libraries/rust/client/src/state/oracles.rs, ./libraries/rust/client/src/state/margin_pool.rs, ./libraries/rust/client/src/wallet.rs, ./libraries/rust/client/src/state.rs, ./libraries/rust/client/src/fixed_term/util.rs, third party dependencies and economic attacks.

REMEDIATION COMMIT ID:

- 5fc06df
- cac3287
- f8a4c15
- c6a322d
- edc38e5
- 4888980
- 84f693d
- 6f054da
- 37370aa
- 2f1a65b
- 29fbb13
- 6ac6f20
- ed33528

- 3842ce3
- f599d38

Out-of-Scope: New features/implementations after the remediation commit IDs.

6. ASSESSMENT SUMMARY & FINDINGS OVERVIEW

CRITICAL	HIGH	MEDIUM	LOW
0	0	1	7

INFORMATIONAL
8

SECURITY ANALYSIS	RISK LEVEL	REMEDIATION DATE
REGISTRY ACCOUNT CAN BE CLOSED BY AN ARBITRARY USER	MEDIUM	SOLVED - 02/19/2025
TWO STEP AUTHORITY TRANSFER NOT IMPLEMENTED	LOW	SOLVED - 02/25/2025
REALLOC RE-INITIALIZATION WASTES COMPUTE UNITS	LOW	SOLVED - 02/21/2025
MISSING MINT EXTENSIONS VALIDATION	LOW	PARTIALLY SOLVED - 03/14/2025
AIRSPACE AUTHORITIES CAN MANIPULATE ARBITRARY METADATA ACCOUNTS	LOW	SOLVED - 03/10/2025
ADMINISTRATOR CAN TRANSFER POSITIONS TO ANOTHER USER	LOW	ACKNOWLEDGED - 03/05/2025

SECURITY ANALYSIS	RISK LEVEL	REMEDIATION DATE
UNVERIFIED PROGRAM PARAMETERS LEADING TO MISCONFIGURATION RISKS	LOW	SOLVED - 03/10/2025
LIQUIDATORS MAY LEAVE LIQUIDATED ACCOUNTS UNHEALTHY	LOW	SOLVED - 03/16/2025
PERMIT REVOCATION OF A REVOKED REGULATOR CANNOT BE DONE BY ANYONE AS INTENDED	INFORMATIONAL	SOLVED - 01/23/2025
FEE OWNER CANNOT WITHRAW FEES	INFORMATIONAL	SOLVED - 03/03/2025
RISK OF UNDEFINED BEHAVIOR DUE TO STACK OVERFLOW	INFORMATIONAL	SOLVED - 02/20/2025
PERMITISSUERREVOKE HELPER FUNCTION USES INCORRECT INSTRUCTION DATA	INFORMATIONAL	SOLVED - 02/28/2025
UNNECESSARY UNSAFE CODE	INFORMATIONAL	SOLVED - 02/28/2025
UNUSED TOKENCONFIG FIELDS RESULT IN INCORRECT TOKEN STANDARD DETERMINATION	INFORMATIONAL	SOLVED - 12/20/2024
RISK OF INCORRECT INTEREST RATE CALCULATION	INFORMATIONAL	SOLVED - 02/28/2025
RISK OF UNHANDLED PANICS	INFORMATIONAL	SOLVED - 03/04/2025

7. FINDINGS & TECH DETAILS

7.1 REGISTRY ACCOUNT CAN BE CLOSED BY AN ARBITRARY USER

// MEDIUM

Description

The `close_registry_account` instruction allows users to close a lookup table registry account, provided the account contains no lookup table entries.

However, this instruction does not verify whether the caller is authorized to close the account, meaning anyone can invoke it. As a result, an unauthorized user could close someone else's account and claim the account's rent.

Additionally, the Lookup Table Registry program does not permit users to reduce the size of a `RegistryAccount` when removing entries. Instead, the rent increases with each reallocation as the account grows. In extreme cases, the rent for a fully allocated account (with a maximum size of 10,272 bytes) could reach approximately 0.07 SOL.

lookup-table-registry/src/lib.rs

```
338 pub struct CloseRegistryAccount<'info> {
339     #[account(mut,
340         close = recipient,
341         constraint = registry_account.len == 0 @ ErrorCode::RegistryNotEmpty
342     )]
343     pub registry_account: Box<Account<'info, RegistryAccount>>,
344
345     /// The authority of the registry account
346     pub authority: Signer<'info>,
347
348     /// The recipient of lamports
349     #[account(mut)]
350     pub recipient: Signer<'info>,
351 }
```

Proof of Concept

1. Initialize registry account with authority A.
2. Close registry account with different signing user.

```
/// Close the registry account
pub fn close_registry_account(_ctx: Context<CloseRegistryAccount>) -> Result<()> {
    msg!(
        "Registry authority: {}",
        _ctx.accounts.registry_account.authority
    );
    msg!("Signing authority: {}", _ctx.accounts.authority.key());
    Ok(())
}
```

```
Program GLoWMgcn3VbyFKiC2FGMgfKxYSyTJS7uKFwKY2CSkq9X consumed 11080 of 200000 compute units
Program GLoWMgcn3VbyFKiC2FGMgfKxYSyTJS7uKFwKY2CSkq9X success
Program GLoWMgcn3VbyFKiC2FGMgfKxYSyTJS7uKFwKY2CSkq9X invoke [1]
Program log: Instruction: InitLookupRegistry
Program LooKUpVskBihZovMhwhEqCER8jwLFHhF4QMZA5axZnJ invoke [2]
Program log: Instruction: InitRegistryAccount
Program 11111111111111111111111111111111 invoke [3]
Program 11111111111111111111111111111111 success
Program LooKUpVskBihZovMhwhEqCER8jwLFHhF4QMZA5axZnJ consumed 8492 of 194729 compute units
Program LooKUpVskBihZovMhwhEqCER8jwLFHhF4QMZA5axZnJ success
Program GLoWMgcn3VbyFKiC2FGMgfKxYSyTJS7uKFwKY2CSkq9X consumed 14111 of 200000 compute units
Program GLoWMgcn3VbyFKiC2FGMgfKxYSyTJS7uKFwKY2CSkq9X success
Program LooKUpVskBihZovMhwhEqCER8jwLFHhF4QMZA5axZnJ invoke [1]
Program log: Instruction: CloseRegistryAccount
Program log: Registry authority: 3QH9HtCpzZ38nJzAfNopQKiJykWPwARjT2utp5XPWzpZ
Program log: Signing authority: 4R2hXfgYJz4xEMGBny2aPZidVa1YBHpZpBavsH2mwjt2
Program LooKUpVskBihZovMhwhEqCER8jwLFHhF4QMZA5axZnJ consumed 25169 of 200000 compute units
Program LooKUpVskBihZovMhwhEqCER8jwLFHhF4QMZA5axZnJ success
```

BVSS

AO:A/AC:L/AX:L/C:N/I:N/A:N/D:M/Y:N/R:N/S:U (5.0)

Recommendation

To address this issue, it is recommended to allow only authorized users (the creator of the registry and eventually the program administrator) to close the registry accounts.

Remediation

SOLVED: The **Glow team** resolved this finding by requiring the `RegistryAccount` authority to sign the instruction.

Remediation Hash

https://github.com/Blueprint-Finance/lookup-table-registry/commit/5fc06df8531e6b635bb86b6182f2d3bbb20d9338

7.2 TWO STEP AUTHORITY TRANSFER NOT IMPLEMENTED

// LOW

Description

The instructions `set_governor` and `airspace_set_authority` allow an authority to set a new authority of the protocol and the airspace, respectively.

However, both instructions pass the new authority is as Public key via instruction parameters and do not require the new authority to sign the instruction. Accidentally setting an incorrect authority would cause loss of control over the protocol or airspace.

set_governor.rs

```
34 pub fn set_governor_handler(ctx: Context<SetGovernor>, new_governor: Pubkey) -> Result<()> {
35     ctx.accounts.governor_id.governor = new_governor;
36
37     Ok(())
38 }
```

airspace_set_authority.rs

```
32 pub fn airspace_set_authority_handler(
33     ctx: Context<AirspaceSetAuthority>,
34     new_authority: Pubkey,
35 ) -> Result<()> {
36     let airspace = &mut ctx.accounts.airspace;
37
38     airspace.authority = new_authority;
39
40     emit!(AirspaceAuthoritySet {
41         airspace: airspace.key(),
42         authority: new_authority
43     });
44
45     Ok(())
46 }
```

BVSS

AO:S/AC:L/AX:L/C:N/I:N/A:C/D:C/Y:N/R:N/S:U (2.5)

Recommendation

To address this issue, it is recommended to require the new authority to sign the instruction as well, ensuring they possess the private key.

Remediation

SOLVED: The **Glow team** resolved this finding by implementing a two-step process. First, an instruction is executed to propose a new authority. Then, a second instruction requires the proposed authority to finalize the transfer by providing its signature.

Remediation Hash

https://github.com/Blueprint-Finance/glow-v1/commit/cac3287b8f9523ba43992753eaa90d3f923eef7f

7.3 REALLOC RE-INITIALIZATION WASTES COMPUTE UNITS

// LOW

Description

The `set_entry` instruction of the Metadata program and the `create_lookup_table` instruction of the Lookup Table Registry program both use the `realloc` function to expand the size of data accounts used by these programs and re-initialize the new memory to zero.

The `realloc` function includes a boolean parameter that determines whether the memory should be re-initialized to zero. Re-initializing memory is helpful for preventing the reuse of old data when the program first decreases the memory size and later increases it again. However, when the account size is only increased, re-initialization is unnecessary because the runtime already provides pre-allocated, zero-filled memory. Performing this re-initialization in such cases results in wasted compute units.

As both instructions can only increase the account data size, memory re-initialization is therefore not necessary.

metadata/src/lib.rs

```
110 pub fn set_entry(ctx: Context<SetEntry>, offset: u64, data: Vec<u8>) -> Result<()> {
111     // Check if the metadata account needs to be resized
112     let metadata_account = ctx.accounts.metadata_account.to_account_info();
113
114     let offset: usize = offset as usize;
115     let data_len = data.len() + offset;
116     let account_len = metadata_account.data_len();
117     if account_len < data_len {
118         // We need to realloc
119         let rent = Rent::get()?;
120         let transfer_amount = rent
121             .minimum_balance(data_len)
122             .saturating_sub(metadata_account.lamports());
123
124         if transfer_amount > 0 {
125             anchor_lang::system_program::transfer(
126                 CpiContext::new(
127                     ctx.accounts.system_program.to_account_info(),
128                     anchor_lang::system_program::Transfer {
129                         from: ctx.accounts.payer.to_account_info(),
130                         to: metadata_account.clone(),
131                     },
132                 ),
133                 transfer_amount,
134             )?;
135         }
136
137         metadata_account.realloc(data_len, true)?;
138     }
```

lookup-table-registry/src/lib.rs

```
184 if append_to_end {
185     // Happy case, add to the end
186     let registry_info = ctx.accounts.registry_account.to_account_info();
187     let existing_len = registry_info.data_len();
188     registry_info.realloc(existing_len + REGISTRY_ENTRY_SIZE, true)?;
189     ctx.accounts.registry_account.tables.push(entry);
190 } else {
```

BVSS

AO:A/AC:L/AX:L/C:N/I:N/A:N/D:L/Y:N/R:N/S:U (2.5)

Recommendation

To address this issue it is recommended to disable the memory re-initialization feature of the `realloc` function when the account data size is only being increased (without first being decreased and then increased again).

Remediation

SOLVED: The **Glow team** solved this finding by disabling the memory re-initialization.

Remediation Hash

https://github.com/Blueprint-Finance/lookup-table-registry/commit/a1ce010d6d8699aeee9a071480b0bf814ba6dec01 https://github.com/Blueprint-Finance/glow-v1/commit/f8a4c1597ea98d11a1b51cb3e6ebd2f3b7ebc5e1

7.4 MISSING MINT EXTENSIONS VALIDATION

// LOW

Description

The Margin Pool Adapter program allows an authority to create token pools using any Token2022 token and to transfer tokens into and out of the pool vault. However, the program does not validate whether the pool vault's mint has any associated extensions, which may introduce security risks.

For example:

- If the mint includes the `TransferFeeConfig` extension, a fee is automatically deducted from each transfer. This means the amount received in the destination pool vault token account will be less than expected, potentially causing data inconsistencies in deposits and loan repayments. The recorded deposit/repayment amount may not match the actual amount transferred.
- If the mint includes the `PermanentDelegate` extension, a designated delegate has unrestricted permissions to transfer and burn tokens from any account associated with that mint, which could lead to unauthorized fund movements.

create_pool.rs

```
87 | /// The mint for the token being custodied by the pool
88 | pub token_mint: Box<InterfaceAccount<'info, Mint>>,
```

BVSS

AO:S/AC:L/AX:L/C:N/I:C/A:N/D:C/Y:N/R:N/S:U (2.5)

Recommendation

To address this issue, it is recommended to validate and restrict the use of mints with potentially harmful extensions.

Remediation

PARTIALLY SOLVED: The **Glow team** partially resolved this issue by validating underlying mints during pool creation and rejecting those with potentially harmful extensions. However, to support the PYUSD (PayPal USD) token, which has the PermanentDelegate extension enabled, this extension needed to remain allowed. To mitigate potential risks, it is necessary to manually review all newly created pools and their underlying mints, ensuring that only mints with the PermanentDelegate extension from trusted third parties are accepted.

Remediation Hash

https://github.com/Blueprint-Finance/glow-v1/commit/c6a322d0311b1a1f546b5025aab65a31a346c31f

7.5 AIRSPACE AUTHORITIES CAN MANIPULATE ARBITRARY METADATA ACCOUNTS

// LOW

Description

The `set_entry` and `remove_entry` instructions in the Metadata program allow airspace authorities to create or delete metadata accounts. However, these instructions do not verify that the provided metadata account has the correct PDA (Program Derived Address). As a result, any airspace authority can modify or remove any metadata account, even if it is not associated with the corresponding airspace.

This issue could lead to metadata manipulation or data inconsistencies within the system.

metadata/src/lib.rs

```
59  #[derive(Accounts)]
60  #[instruction(len: u16)]
61  pub struct SetEntry<'info> {
62      /// The address paying the rent for the account if additional rent is required
63      #[account(mut)]
64      pub payer: Signer<'info>,
65
66      /// The account containing the metadata to change
67      /// CHECK:
68      #[account(mut)]
69      pub metadata_account: AccountInfo<'info>,
70
71      /// The authority that must sign to make this change
72      pub authority: Signer<'info>,
73
74      /// The airspace that the entry belongs to]
75      #[cfg_attr(not(feature = "testing"), account(
76          constraint = airspace.authority == authority.key(),
77      ))]
78      pub airspace: Box<Account<'info, Airspace>>,
79
80      pub system_program: Program<'info, System>,
81  }
82
83  #[derive(Accounts)]
84  pub struct RemoveEntry<'info> {
85      /// The account containing the metadata to change
86      /// CHECK: This is safe because we can only mutate accounts that we own, and we are
87      /// closing this account. The metadata program only has entries, so this is then
88      /// presumed to always be an entry. The risk could be if we close a different type
89      /// of account unintentionally.
90      #[account(mut)]
91      pub metadata_account: AccountInfo<'info>,
92
93      /// The authority that must sign to make this change
94      pub authority: Signer<'info>,
95
96      /// The airspace that the entry belongs to
97      #[cfg_attr(not(feature = "testing"), account(
98          constraint = airspace.authority == authority.key(),
99      ))]
100     pub airspace: Box<Account<'info, Airspace>>,
101
102     /// The address receiving the rent
103     /// CHECK: Some details
104     #[account(mut)]
105     pub receiver: AccountInfo<'info>,
106 }
```

Proof of Concept

1. Create two margin pools: poolA and poolB with their associated airspaces (airspace1 and airspace2) and different airspace authorities (authority1 and autority2).
2. Use airspace authority1 to remove a metadata account associated with airspace2.

```
println!("airspace 1 = {}", ctx.airspace_details.address);
println!("airspace 2 = {}", ctx2.airspace_details.address);
let pool1: MarginPoolIxBuilder = MarginPoolIxBuilder::new(ctx.airspace_details.address, env.usdc);
let pool2: MarginPoolIxBuilder = MarginPoolIxBuilder::new(ctx2.airspace_details.address, env2.usdc);
let metadata_account: Pubkey = get_metadata_address(&pool1.airspace, &pool1.token_mint.address);
println!( "metadata_account 1 = {} set by authority = {}", metadata_account, ctx.airspace_authority.pubkey() );
let data: Vec<u8> = glow_metadata::instruction::SetEntry { offset: 0, data: vec![0], } .data();

let ix: Instruction = Instruction {
    program_id: glow_metadata::ID,
    data,
    accounts: glow_metadata::accounts::SetEntry {
        payer: ctx.airspace_authority.pubkey(),
        metadata_account,
        authority: ctx.airspace_authority.pubkey(),
        airspace: pool1.airspace,
        system_program: anchor_lang::system_program::ID,
    }
    .to_account_metas(None),
};
let tx: Signature = ix .clone() .with_signer(&ctx.airspace_authority) .send_and_confirm(&ctx.rpc()) .await?;

println!( "metadata_account 1 = {} removed by authority = {}", metadata_account, ctx2.airspace_authority.pubkey() );
let data2: Vec<u8> = glow_metadata::instruction::RemoveEntry {}.data();
let ix2: Instruction = Instruction {
    program_id: glow_metadata::ID,
    data: data2,
    accounts: glow_metadata::accounts::RemoveEntry {
        metadata_account,
        authority: ctx2.airspace_authority.pubkey(),
        airspace: pool2.airspace,
        receiver: ctx2.airspace_authority.pubkey(),
    }
    .to_account_metas(None),
};
let tx: Signature = ix2 .with_signer(&ctx2.airspace_authority) .send_and_confirm(&ctx2.rpc()) .await?;
```

```
airspace 1 = 6LNAvN4DxVUtQZ9RoZyPfADoDDHLND6MyxMYpRW9PP71
airspace 2 = Dc39fkt2HM6JPDF8DGY2xSDKFHv56BLn4mXTTBFsGJvw
metadata_account 1 = GViD1BgiWqWn4LQpH8cZSG9ggY3Kkcgb8BerYK9wu4QG set by authority = CoKapDF2p4fQcbPfG61KaASEkpCATqZ41FegrVqtnz4j
[2025-02-05T11:59:57.919953000Z DEBUG solana_runtime::message_processor::stable_log] Program yT2ut38wC6A6zsGo2aUgy9kkh8EBuNXYvtmo7aUg1oW invoke [1]
[2025-02-05T11:59:57.920336000Z DEBUG solana_runtime::message_processor::stable_log] Program log: Instruction: SetEntry
[2025-02-05T11:59:57.920571000Z DEBUG solana_runtime::message_processor::stable_log] Program yT2ut38wC6A6zsGo2aUgy9kkh8EBuNXYvtmo7aUg1oW consumed 2348 of 200000 compute units
[2025-02-05T11:59:57.920590000Z DEBUG solana_runtime::message_processor::stable_log] Program yT2ut38wC6A6zsGo2aUgy9kkh8EBuNXYvtmo7aUg1oW success
metadata_account 1 = GViD1BgiWqWn4LQpH8cZSG9ggY3Kkcgb8BerYK9wu4QG removed by authority = FC6aGWCfEgjhsqp3F2iBoHh4hmb2BJSj8f3KZeqKjiYm
[2025-02-05T11:59:57.922651000Z DEBUG solana_runtime::message_processor::stable_log] Program yT2ut38wC6A6zsGo2aUgy9kkh8EBuNXYvtmo7aUg1oW invoke [1]
[2025-02-05T11:59:57.922969000Z DEBUG solana_runtime::message_processor::stable_log] Program log: Instruction: RemoveEntry
[2025-02-05T11:59:57.923210000Z DEBUG solana_runtime::message_processor::stable_log] Program yT2ut38wC6A6zsGo2aUgy9kkh8EBuNXYvtmo7aUg1oW consumed 2182 of 200000 compute units
[2025-02-05T11:59:57.923230000Z DEBUG solana_runtime::message_processor::stable_log] Program yT2ut38wC6A6zsGo2aUgy9kkh8EBuNXYvtmo7aUg1oW success
[2025-02-05T11:59:57.925710000Z DEBUG solana_runtime::message_processor::stable_log] Program yT2ut38wC6A6zsGo2aUgy9kkh8EBuNXYvtmo7aUg1oW invoke [1]
[2025-02-05T11:59:57.926097000Z DEBUG solana_runtime::message_processor::stable_log] Program log: Instruction: SetEntry
[2025-02-05T11:59:57.926336000Z DEBUG solana_runtime::message_processor::stable_log] Program yT2ut38wC6A6zsGo2aUgy9kkh8EBuNXYvtmo7aUg1oW consumed 2348 of 200000 compute units
[2025-02-05T11:59:57.926357000Z DEBUG solana_runtime::message_processor::stable_log] Program yT2ut38wC6A6zsGo2aUgy9kkh8EBuNXYvtmo7aUg1oW success
```

BVSS

AO:S/AC:L/AX:L/C:N/I:C/A:N/D:N/Y:N/R:N/S:U (2.5)

Recommendation

To address this issue it is recommended to verify the address of the metadata account and ensure that it is associated with the expected airspace.

Remediation

SOLVED: The **Glow team** solved this finding by verifying that the provided metadata account is correctly linked to the specified airspace using the appropriate seeds for the metadata's PDA.

Remediation Hash

https://github.com/Blueprint-Finance/glow-v1/commit/edc38e5f53c5d1f275ff2d2bd34da3bd6e85aacc

7.6 ADMINISTRATOR CAN TRANSFER POSITIONS TO ANOTHER USER

// LOW

Description

The `admin_transfer_position` instruction enables the protocol administrator to transfer any position from one `MarginAccount` to another.

According to the documentation, this functionality is intended as a mechanism for manually resolving issues in the protocol caused by problematic user assets.

However, the instruction does not verify that both `MarginAccounts` belong to the same owner. As a result, the administrator could potentially transfer positions between accounts owned by different users, which could lead to the loss of user funds.

admin_transfer_position.rs

```
33  pub struct AdminTransferPosition<'info> {
34      /// The administrative authority
35      #[account(address = GOVERNOR_ID)]
36      pub authority: Signer<'info>,
37
38      /// The target margin account to move a position into
39      #[account(mut)]
40      pub target_account: AccountLoader<'info, MarginAccount>,
41
42      /// The source account to move a position out of
43      #[account(mut)]
44      pub source_account: AccountLoader<'info, MarginAccount>,
45
46      /// The token account to be moved from
47      #[account(mut, token::mint = token_mint, token::token_program = token_program)]
48      pub source_token_account: InterfaceAccount<'info, TokenAccount>,
49
50      /// The token account to be moved into
51      #[account(mut, token::mint = token_mint)]
52      pub target_token_account: InterfaceAccount<'info, TokenAccount>,
53
54      pub token_mint: InterfaceAccount<'info, Mint>,
55
56      pub token_program: Interface<'info, TokenInterface>,
57  }
```

BVSS

AO:S/AC:L/AX:L/C:N/I:N/A:N/D:C/Y:N/R:N/S:U (2.0)

Recommendation

To address this issue, it is recommended to ensure that both the source and target `MarginAccounts` belong to the same owner. This safeguard will prevent accidental transfers between accounts owned by different users.

Remediation

ACKNOWLEDGED: The **Glow team** acknowledged this finding but chose to maintain the existing implementation. The instruction is intended for scenarios where an account cannot be liquidated

or in rare, extreme circumstances.

7.7 UNVERIFIED PROGRAM PARAMETERS LEADING TO MISCONFIGURATION RISKS

// LOW

Description

The `configure_token` instruction allows the airspace authority to set and update token configurations. However, the parameters `value_modifier`, `max_staleness, admin` and `token_kind` are not validated, leaving them open to being set to arbitrary values:

- `max_staleness`: If this parameter is incorrectly set to 0 or an excessively large value, it will effectively disable the program's staleness verification mechanism.
- `value_modifier`: If this parameter is set to a value greater than 100, it could enable the issuance of under-collateralized loans.
- `token_kind`: Accidentally updating the token kind from Claim to Collateral or vice versa would corrupt the data integrity of the protocol that might cause losses for the protocol or for the users.
- `admin`: Accidentally setting an incorrect account as token administrator could compromise the token price results. In addition the instruction `refresh_deposit_position` does not verify the correct owner of the Pyth oracle price account which could be an issue in case the oracle is set incorrectly during token configuration.

configure_token.rs

```
78  pub fn configure_token_handler(
79      ctx: Context<ConfigureToken>,
80      updated_config: Option<TokenConfigUpdate>,
81  ) -> Result<()> {
82      let config = &mut ctx.accounts.token_config;
83
84      emit!(TokenConfigured {
85          airspace: ctx.accounts.airspace.key(),
86          mint: ctx.accounts.mint.key(),
87          update: updated_config.clone(),
88      });
89
90      let updated_config = match updated_config {
91          Some(update) => update,
92          None => return config.close(ctx.accounts.payer.to_account_info()),
93      };
94
95      if config.underlying_mint != Pubkey::default()
96          && updated_config.underlying_mint != config.underlying_mint
97      {
98          msg!("underlying mint cannot be changed");
99          return err!(ErrorCode::InvalidConfig);
100     }
101
102     config.mint = ctx.accounts.mint.key();
103     config.airspace = ctx.accounts.airspace.key();
104     config.underlying_mint = updated_config.underlying_mint;
105     config.admin = updated_config.admin;
106     config.token_kind = updated_config.token_kind;
107     config.value_modifier = updated_config.value_modifier;
108     config.max_staleness = updated_config.max_staleness;
109
110     config.validate()?;
111
112     Ok(())
113 }
```

The `configure` margin pool instruction allows the airspace authority to set and update pool configurations. However non of the `MarginPoolConfig` parameters are validated, leaving them open to being set to arbitrary values.

The program does not guarantee, that the parameters `utilization_rate_*` and `borrow_rate_*` are set with increasing values which could cause unexpected behavior. The parameter `management_fee_rate` is not capped to 10000 and setting it to greater value could result in incorrect fee and interest calculations.

configure.rs

```
126 pub fn configure_handler(
127     ctx: Context<Configure>,
128     metadata: Option<TokenMetadataParams>,
129     config: Option<MarginPoolConfig>,
130     oracle: Option<TokenPriceOracle>,
131 ) -> Result<()> {
132     let pool = &mut ctx.accounts.margin_pool;
133 
134     if let Some(new_config) = config {
135         pool.config = new_config;
136     }
```

Proper validation of these parameters is essential to ensure the program functions as intended and avoids potential misuse.

BVSS

AO:S/AC:L/AX:L/C:N/I:N/A:N/D:C/Y:N/R:N/S:U (2.0)

Recommendation

To address this issue, it is recommended to validate the `value_modifier` and `max_staleness` parameters by restricting their values to predefined acceptable ranges. The `token_kind` parameter must be set only during the initial token configuration and cannot not be allowed to be modified afterward. When setting the `admin` parameter, the program must verify that the oracle price account has correct data format and is owned by the Pyth program.
Validate the `MarginPoolConfig` parameters of the `configure` instruction and make sure their values are in expected ranges.

Remediation

SOLVED: The **Glow team** solved this finding by adding validation checks to the parameters of the `configure_token` and `configure` instructions, ensuring they can only be updated with expected and meaningful values. Additionally, the oracle price account implementation was refactored, making the validation of the price account in the admin parameter unnecessary.

Remediation Hash

https://github.com/Blueprint-Finance/glow-v1/commit/a8ab41fd603e00edaae6b0a68ad919167373d8a9 https://github.com/Blueprint-Finance/glow-v1/commit/ceecb194701fb3c7d6efe846cbb8cf1511d7ed9a https://github.com/Blueprint-Finance/glow-v1/commit/4888980a14b92e9314f608e951285089d1f2ea5d

7.8 LIQUIDATORS MAY LEAVE LIQUIDATED ACCOUNTS UNHEALTHY

// LOW

Description

The instructions `liquidate_begin`, `liquidator_invoke` and `liquidate_end` are used by the authorized liquidators to initiate, execute and end liquidations of unhealthy accounts. However the liquidation process does not guarantee that a liquidated account will be healthy after the liquidation ends. A liquidator can liquidate an account only partially, take a liquidation fee and leave the account unhealthy. This process may be repeated until the liquidated account will be drained.

liquidate_end.rs

```
45  pub fn liquidate_end_handler(ctx: Context<LiquidateEnd>) -> Result<()> {
46      let mut account = ctx.accounts.margin_account.load_mut()?;
47      let start_time = ctx.accounts.liquidation.load()?.state.start_time();
48
49      let timed_out = Clock::get()?.unix_timestamp - start_time >= LIQUIDATION_TIMEOUT;
50
51      if (account.liquidator != ctx.accounts.authority.key()) && !timed_out {
52          msg!(
53              "Only the liquidator may end the liquidation before the timeout of {} seconds",
54              LIQUIDATION_TIMEOUT
55          );
56          return Err(ErrorCode::UnauthorizedLiquidator.into());
57      }
58
59      account.end_liquidation();
60
61      emit!(events::LiquidationEnded {
62          margin_account: ctx.accounts.margin_account.key(),
63          authority: ctx.accounts.authority.key(),
64          timed_out,
65      });
66
67      Ok(())
68  }
```

margin/src/state/account.rs

```
161 pub fn end_liquidation(&mut self) {
162     self.liquidator = Pubkey::default();
163 }
```

BVSS

AO:S/AC:L/AX:L/C:N/I:N/A:N/D:C/Y:N/R:N/S:U (2.0)

Recommendation

To address this issue, it is recommended to incentivize liquidators to fully resolve unhealthy accounts by allowing them to collect liquidation fees only after the account has been restored to a healthy state.

Remediation

SOLVED: The **Glow team** resolved this issue by introducing a mechanism that accumulates the liquidator's fees during the liquidation process. Liquidators can only collect their fees once the account has been restored to a healthy state.

Remediation Hash

https://github.com/Blueprint-Finance/glow-v1/commit/84f693d9949ec1310ca2e2d070374b4839e3e22c

7.9 PERMIT REVOCATION OF A REVOKED REGULATOR CANNOT BE DONE BY ANYONE AS INTENDED

// INFORMATIONAL

Description

The instruction `airspace_permit_revoke` allows to revoke permits. The intended behavior based on the documentation is that "For restricted airspaces, anyone can revoke a permit from a revoked regulator.".

However the instruction incorrectly requires the signature of the airspace authority or a permit issuer and thus does not allow any user to revoke the permit of a revoked regulator.

airspace_permit_revoke.rs

```
69 | // The airspace authority or issuing regulator is always allowed to revoke
70 | if authority != airspace.authority && authority != permit.issuer {
71 |     return err!(AirspaceErrorCode::PermissionDenied);
72 | }
```

BVSS

AO:S/AC:L/AX:L/C:N/I:N/A:M/D:M/Y:N/R:N/S:U (1.3)

Recommendation

To address this issue, it is recommended to re-evaluate the requirement stating that "for restricted airspaces, anyone can revoke a permit from a revoked regulator." If this requirement has a valid business justification, the implementation of the `airspace_permit_revoke` instruction should be adjusted accordingly. However, it is important to consider that allowing unrestricted revocation would enable anyone to close the affected permit accounts and claim the associated rent fees.

Remediation

SOLVED: The **Glow team** resolved this finding by aligning the code to the documentation and allowing anyone to revoke a permit if an airspace is restricted and the issuer is revoked.

Remediation Hash

https://github.com/Blueprint-Finance/glow-v1/commit/6f054da857af8b709eec976f67ce43cf7c400564

7.10 FEE OWNER CANNOT WITHRAW FEES

// INFORMATIONAL

Description

The `withdraw_fees` instruction is intended to allow the fee owner to withdraw collected fees. However, it consistently fails due to an incorrect token account authority during the burning of deposit notes. Currently, the authority is set to `margin_pool`, but it should be set to `fee_owner` for the instruction to execute successfully.

withdraw_fees.rs

```
87  fn burn_note_context(&self) -> CpiContext<'_, '_, '_, 'info, Burn<'info>> {
88      CpiContext::new(
89          self.pool_token_program.to_account_info(),
90          Burn {
91              from: self.fee_destination.to_account_info(),
92              mint: self.deposit_note_mint.to_account_info(),
93              // authority: self.margin_pool.to_account_info(),
94              authority: self.fee_owner.to_account_info(),
95          },
96      )
97  }
```

BVSS

AO:S/AC:L/AX:L/C:N/I:N/A:C/D:M/Y:M/R:F/S:U (0.6)

Recommendation

To address this issue, it is recommended to update the deposit notes authority to `fee_owner` when burning tokens.

Remediation

SOLVED: The **Glow team** resolved this finding by setting the correct authority when burning tokens.

Remediation Hash

https://github.com/Blueprint-Finance/glow-v1/commit/37370aa97b8d004342e666f3d1bf32fb9d8afda1

7.11 RISK OF UNDEFINED BEHAVIOR DUE TO STACK OVERFLOW

// INFORMATIONAL

Description

The margin pool instruction `create_pool` uses too many stack allocated variables which causes the following error during program compilation:

```
Compiling glow-margin-pool v1.0.0 (./glow-v1/programs/margin-pool)
Error: Function
_ZN165_$LT$glow_margin_pool..instructions..create_pool..CreatePool$u20$as$u20$a
nchor_lang..Accounts$LT$glow_margin_pool..instructions..create_pool..CreatePool
Bumps$GT$$GT$12try_accounts17ha2829e083fe80a7eE Stack offset of 6896 exceeded
max offset of 4096 by 2800 bytes, please minimize large stack variables
```

Stack overflow can lead to undefined behavior and currently blocks deployment to devnet and mainnet. This issue arises due to multiple factors:

1. **Anchor 0.30.1 overuses stack variables** – See this issue.
2. **Differences in feature activation** – Devnet and mainnet lack certain features available on localnet, as detailed here.

BVSS

AO:S/AC:L/AX:L/C:N/I:M/A:C/D:N/Y:N/R:F/S:U (0.6)

Recommendation

To address this issue, it is recommended to reduce the size of stack variables by using the `Box` type and allocating variables on the heap, using `zero_copy` or splitting instructions into multiple smaller instructions.

Remediation

SOLVED: The **Glow team** resolved this issue by manually initializing accounts instead of using Anchor's `init` constraint, effectively reducing stack usage.

Remediation Hash

https://github.com/Blueprint-Finance/glow-v1/commit/2f1a65b195b76e6076e03771b868f8791c74dae8

7.12 PERMITISSUERREVOKE HELPER FUNCTION USES INCORRECT INSTRUCTION DATA

// INFORMATIONAL

Description

The library function `permit_issuer_revoke` is a helper function to create the `AirspacePermitIssuerRevoke` instruction. However this function accidentally creates an instruction with `AirspacePermitIssuerCreate` data instead of `AirspacePermitIssuerRevoke` data.

Sending such malformed instruction will cause the instruction to fail. This issue does not have any security implications, but may cause problems for example during testing where the tests will unexpectedly fail.

airspace.rs

```
185 pub fn permit_issuer_revoke(&self, issuer: Pubkey) -> Instruction {
186     let accounts = glow_airspace::accounts::AirspacePermitIssuerRevoke {
187         airspace: self.address,
188         authority: self.authority,
189         receiver: self.payer,
190         issuer_id: self.derive_issuer_id(&issuer),
191     }
192     .to_account_metas(None);
193 
194     Instruction {
195         accounts,
196         program_id: glow_airspace::ID,
197         data: glow_airspace::instruction::AirspacePermitIssuerCreate { issuer }.data(),
198     }
199 }
```

BVSS

AO:S/AC:L/AX:L/C:N/I:N/A:N/D:N/Y:N/R:F/S:U (0.0)

Recommendation

To address this issue, it is recommended to change the `AirspacePermitIssuerCreate` data to `AirspacePermitIssuerRevoke`.

Remediation

SOLVED: The **Glow team** resolved this finding by setting the correct instruction data.

Remediation Hash

https://github.com/Blueprint-Finance/glow-v1/commit/29fbb13ce066632a2cd386912dbe5677b28ce83d

7.13 UNNECESSARY UNSAFE CODE

// INFORMATIONAL

Description

The instruction `AirspacePermitCreate` uses unsafe code to disable lifetime checker. However the use of unsafe code is not necessary in this case. Unsafe code in Solana programs is considered bad practice and should be avoided in order to prevent unexpected behavior.

airspace_permit_create.rs

```
79 // If the airspace is not restricted, then any signer can create permits
80 if airspace.is_restricted && airspace.authority != authority.key() {
81     // For a restricted airspace, the optional regulator account needs to be verified
82     // to prove that the signer is authorized to create the permit
83     let _ = Account::<AirspacePermitIssuerId>::try_from(unsafe {
84         // 2024-11-05: The transmute is used to extend lifetimes, as it's otherwise been impossible to solve the lifetime
85         // constraints here.
86         // This is expected to be resolved in a future Anchor version: https://github.com/coral-xyz/anchor/pull/3340
87         std::mem::transmute::<&AccountInfo<'_>, &AccountInfo<'_>>(&ctx.accounts.issuer_id)
88     })?;
89
90     // No further checks are necessary, since the address is already verified by anchor,
91     // and the account data being valid to deserialize means the permission was granted
92 }
```

BVSS

AO:S/AC:L/AX:L/C:N/I:N/A:N/D:N/Y:N/R:F/S:U (0.0)

Recommendation

To address this issue, it is recommended to replace the unsafe code by a safe alternative such as: `let = AirspacePermitIssuerId::try_deserialize( &mut &ctx.accounts.issuer_id.data.try_borrow().unwrap()[..], );`

This code is safe and will verify that the `issuer_id` account was correctly initialized as expected.

Remediation

SOLVED: The **Glow team** resolved this finding by removing the unsafe code and implementing the recommended solution.

Remediation Hash

https://github.com/Blueprint-Finance/glow-v1/commit/6ac6f204aaa0ea3c7195f430f3973277d8af05d4

7.14 UNUSED TOKENCONFIG FIELDS RESULT IN INCORRECT TOKEN STANDARD DETERMINATION

// INFORMATIONAL

Description

The `configure_tokens` instruction allows adding new token configurations or updating existing ones in the tokens whitelist. However, this instruction does not explicitly set the `TokenConfig::mint_token_program` and `TokenConfig::underlying_mint_token_program` fields, leaving them at their default values.

As a result, when a user registers a new position using the `register_position` instruction, the program incorrectly determines the associated token standard. The program compares the `TokenConfig::mint_token_program` or `TokenConfig::underlying_mint_token_program` addresses based on the type of asset and checks if they match the address of the Token 2022 program. If they do not match, the `AccountPosition::is_token_2022` field is set to `0`, indicating that the legacy token program is being used. Since the `TokenConfig` program addresses are never set, the `is_token_2022` field is always set to `0`, even when the Token 2022 program was actually used.

While this issue does not pose a security risk, it breaks the existing client code and helper functions that interact with the protocol.

configure_token.rs

```
102 config.mint = ctx.accounts.mint.key();
103 config.airspace = ctx.accounts.airspace.key();
104 config.underlying_mint = updated_config.underlying_mint;
105 config.admin = updated_config.admin;
106 config.token_kind = updated_config.token_kind;
107 config.value_modifier = updated_config.value_modifier;
108 config.max_staleness = updated_config.max_staleness;
```

positions.rs

```
449 pub fn new_from_config(
450     config: &Account<TokenConfig>,
451     mint_decimals: u8,
452     address: Pubkey,
453     adapter: Pubkey,
454 ) -> Self {
455     Self {
456         mint: config.mint,
457         token_program: match config.token_kind {
458             TokenKind::Collateral => config.underlying_mint_token_program,
459             TokenKind::Claim => config.mint_token_program,
460             TokenKind::AdapterCollateral => config.mint_token_program,
461         },
```

account.rs

```
216 if let Some(free_position) = free_position {
217     free_position.exponent = -(config.decimals as i16);
218     free_position.address = config.address;
219     free_position.adapter = config.adapter;
220     free_position.kind = config.kind.into_integer();
221     free_position.balance = 0;
222
```

```
223     free_position.value_modifier = config.value_modifier;
224     free_position.max_staleness = config.max_staleness;
225     // NIT: This isn't a great way of indicating token support, because what happens if
226     // there is token_2026 in future?
227     free_position.is_token_2022 = if config.token_program == anchor_spl::token_2022::ID {
228         1
229     } else {
230         0
        };
```

BVSS

AO:S/AC:L/AX:L/C:N/I:N/A:N/D:N/Y:N/R:F/S:U (0.0)

Recommendation

To address this issue, it is recommended to correctly set the `TokenConfig::mint_token_program` and `TokenConfig::underlying_mint_token_program` fields during token configuration in the `token_configure` instruction. This will enable correct determination of the token program during position registration.

Remediation

SOLVED: The **Glow team** resolved this finding by correctly setting the respective `TokenConfig` fields.

Remediation Hash

https://github.com/Blueprint-Finance/glow-v1/commit/ed335283ede3416f8338069edfaa47e5ead3fb17

7.15 RISK OF INCORRECT INTEREST RATE CALCULATION

// INFORMATIONAL

Description

The `MarginPool::interest_rate` method calculates the current interest rate for loans. It also handles the edge case where the pool is empty and contains zero deposit notes.

Currently, when the pool is empty, the method returns `borrow_rate_1`, which represents the transition point between the first and second interest rate regimes. However, the correct rate in this scenario should be `borrow_rate_0`, as it corresponds to the starting point of the first regime.

Although this does not currently cause incorrect results—since the interest rate is multiplied by zero when the pool is empty—it should still be corrected to prevent potential issues in future code updates.

margin-pool/src/state.rs

```
296 pub fn interest_rate(&self) -> Number {
297     let borrow_1 = Number::from_bps(self.config.borrow_rate_1);
298
299     // Catch the edge case of empty pool
300     if self.deposit_notes == 0 {
301         return borrow_1;
302     }
```

BVSS

AO:A/AC:L/AX:L/C:N/I:N/A:N/D:N/Y:N/R:N/S:U (0.0)

Recommendation

To address this issue it is recommended to return the `borrow_rate_0` interest rate when the pool is empty.

Remediation

SOLVED: The **Glow team** resolved this issue by returning the `borrow_rate_0` interest rate when the pool is empty.

Remediation Hash

https://github.com/Blueprint-Finance/glow-v1/commit/3842ce38824adb52d5e93ca180ed1a728f1ce6b2

7.16 RISK OF UNHANDLED PANICS

// INFORMATIONAL

Description

The `liquidate_begin` instruction allows a liquidator to initiate a liquidation. As part of its logic, the instruction checks whether the account is already undergoing liquidation.

If the same liquidator attempts to invoke this instruction twice, the program will panic by calling the `unreachable!()` macro. This macro is typically intended for cases where the compiler cannot infer that a specific branch of code should never be reached.

However, in this scenario, it is entirely possible for the same liquidator to call `liquidate_begin` twice. Therefore, using the `unreachable!()` macro is inappropriate.

liquidate_begin.rs

```
76  // verify not already being liquidated
77  match account.liquidator {
78      liq if liq == liquidator => {
79          // this liquidator has already been set as the active liquidator,
80          // so there is nothing to do
81          unreachable!();
82      }
83
84      liq if liq == Pubkey::default() => {
85          // not being liquidated, so claim it
86          account.start_liquidation(liquidator);
87      }
88
89      _ => {
90          // already claimed by some other liquidator
91          return Err(ErrorCode::Liquidating.into());
92      }
93  }
```

BVSS

AO:A/AC:L/AX:L/C:N/I:N/A:N/D:N/Y:N/R:N/S:U (0.0)

Recommendation

To resolve this issue, it is recommended to handle cases where the same liquidator invokes the `liquidate_begin` instruction twice by either treating it as a no-op or returning a meaningful error message.

Remediation

SOLVED: The **Glow team** resolved this finding by ensuring that multiple invocations of the `liquidate_begin` instruction by the same liquidator have no effect.

Remediation Hash

https://github.com/Blueprint-Finance/glow-v1/commit/f599d38331636c6235167147c4071dc83a86e74a

8. AUTOMATED TESTING

STATIC ANALYSIS REPORT

Description

Halborn used automated security scanners to assist with detection of well-known security issues and vulnerabilities. Among the tools used was `cargo audit`, a security scanner for vulnerabilities reported to the RustSec Advisory Database. All vulnerabilities published in `https://crates.io` are stored in a repository named The RustSec Advisory Database. `cargo audit` is a human-readable version of the advisory database which performs a scanning on Cargo.lock. Security Detections are only in scope. All vulnerabilities shown here were already disclosed in the above report. However, to better assist the developers maintaining this code, the auditors are including the output with the dependencies tree, and this is included in the cargo audit output to better know the dependencies affected by unmaintained and vulnerable crates.

Cargo Audit Results

ID	CRATE	DESCRIPTION
RUSTSEC-2024-0344	curve25519-dalek	Timing variability in `curve25519-dalek`'s `Scalar29::sub`/`Scalar52::sub`
RUSTSEC-2022-0093	ed25519-dalek	Double Public Key Signing Function Oracle Attack on `ed25519-dalek`

Halborn strongly recommends conducting a follow-up assessment of the project either within six months or immediately following any material changes to the codebase, whichever comes first. This approach is crucial for maintaining the project's integrity and addressing potential vulnerabilities introduced by code modifications.