HALBORN

Mercurial Finance – Vault and Dynamic AMM

Solana Program Security Audit

Prepared by: **Halborn**

Date of Engagement: **June 27th, 2022 - July 25th, 2022**

Visit: **Halborn.com**

DOCUMENT REVISION HISTORY

VERSION	MODIFICATION	DATE	AUTHOR
0.1	Document Creation	07/25/2022	Pablo Gomez
0.2	Final Draft	07/26/2022	Pablo Gomez
0.3	Draft Review	07/26/2022	Gabi Urrutia
1.0	Remediation Plan	08/02/2022	Pablo Gómez
1.1	Remediation Plan Review	08/02/2022	Gabi Urrutia

CONTACTS

CONTACT	COMPANY	EMAIL
Rob Behnke	Halborn	Rob.Behnke@halborn.com
Steven Walbroehl	Halborn	Steven.Walbroehl@halborn.com
Gabi Urrutia	Halborn	Gabi.Urrutia@halborn.com
Pablo Gomez	Halborn	Pablo.Gomez@halborn.com

EXECUTIVE OVERVIEW

1.1 INTRODUCTION

Mercurial Finance engaged Halborn to conduct a security audit on their Solana programs, beginning on June 27th, 2022 and ending on July 25th, 2022 . The security assessment was scoped to the programs provided in the Mercurial GitHub repository. Commit hashes and further details can be found in the Scope section of this report.

1.2 AUDIT SUMMARY

The team at Halborn was provided five weeks for the engagement and assigned a full-time security engineer to audit the security of the programs in scope. The security engineer is a blockchain and Solana program security expert with advanced penetration testing and Solana program hacking skills, and deep knowledge of multiple blockchain protocols.

The purpose of this audit is to:

- Identify potential security issues within the programs

In summary, Halborn identified some improvements to reduce the likelihood and impact of multiple risks, which have been partially addressed by Mercurial Finance . The main ones are the following:

- Avoid using potential insecure custom masking functions with known collisions
- Avoid using helper methods like unwrap() that can lead to a potential DoS

1.3 TEST APPROACH & METHODOLOGY

Halborn performed a combination of manual review of the code and automated security testing to balance efficiency, timeliness, practicality, and

accuracy in regard to the scope of the program audit. While manual testing is recommended to uncover flaws in logic, process, 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 audit:

- Research into the architecture, purpose, and use of the platform.
- Manual program code review and walkthrough to identify logic issues.
- Mapping out possible attack vectors
- Thorough assessment of safety and usage of critical Rust variables and functions in scope that could led to arithmetic vulnerabilities.
- Finding unsafe Rust code usage (cargo-geiger)

- Scanning dependencies for known vulnerabilities (cargo audit).
- Local runtime instance testing (solana-test-framework)
- Scanning for common Solana vulnerabilities (soteria)

RISK METHODOLOGY:

Vulnerabilities or issues observed by Halborn are ranked based on the risk assessment methodology by measuring the **LIKELIHOOD** of a security incident and the **IMPACT** should an incident occur. This framework works for communicating the characteristics and impacts of technology vulnerabilities. The quantitative model ensures repeatable and accurate measurement while enabling users to see the underlying vulnerability characteristics that were used to generate the Risk scores. For every vulnerability, a risk level will be calculated on a scale of 5 to 1 with 5 being the highest likelihood or impact.

RISK SCALE - LIKELIHOOD

5 - Almost certain an incident will occur.
4 - High probability of an incident occurring.
3 - Potential of a security incident in the long term.
2 - Low probability of an incident occurring.
1 - Very unlikely issue will cause an incident.

RISK SCALE - IMPACT

5 - May cause devastating and unrecoverable impact or loss.
4 - May cause a significant level of impact or loss.
3 - May cause a partial impact or loss to many.
2 - May cause temporary impact or loss.
1 - May cause minimal or un-noticeable impact.

The risk level is then calculated using a sum of these two values, creating a value of 10 to 1 with 10 being the highest level of security risk.

10 - CRITICAL
9 - 8 - HIGH
7 - 6 - MEDIUM
5 - 4 - LOW
3 - 1 - VERY LOW AND INFORMATIONAL

1.4 SCOPE

Code repositories:

1. Mercurial Vault

   - Repository: mercurial-vault
   - Commit ID: d901a61ccde894517692236347bd324c44d1b983
   - Programs in scope:
     1. vault (programs/vault)

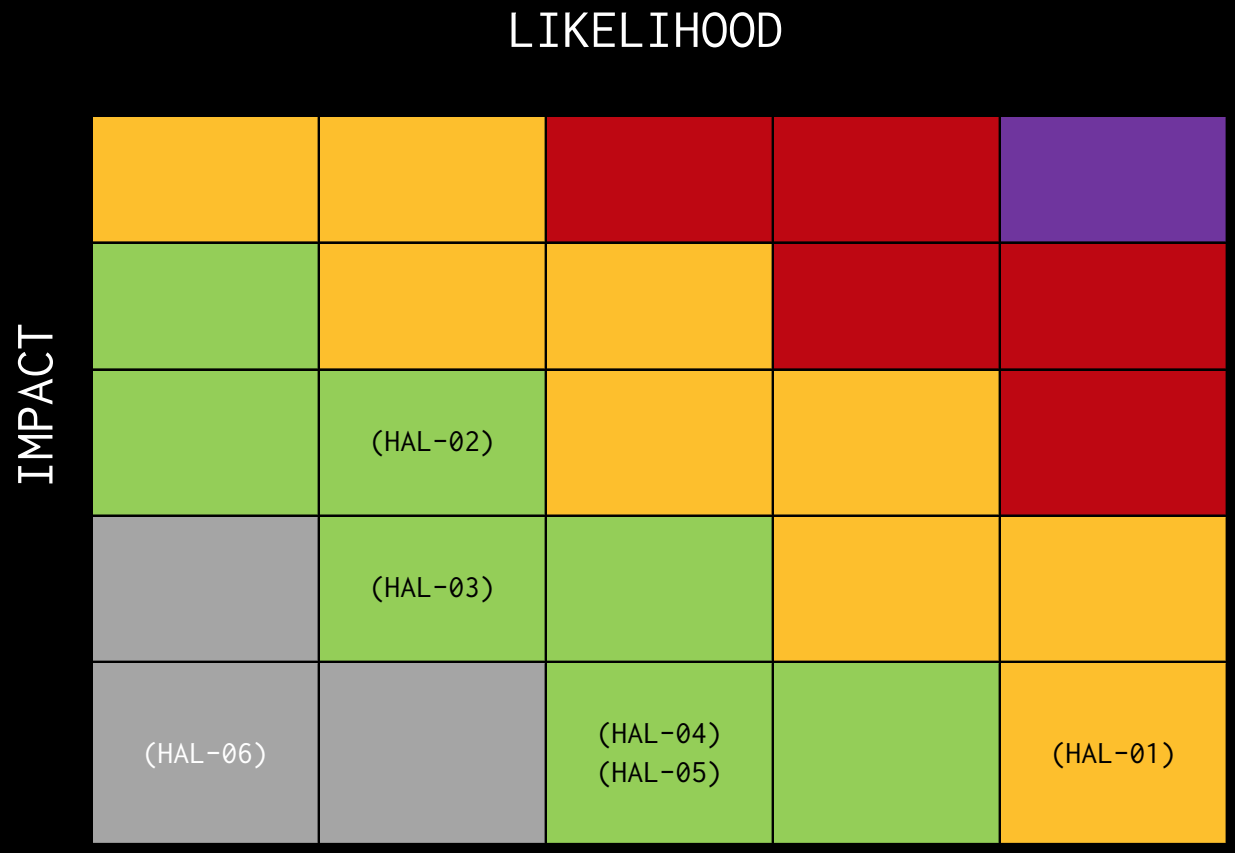
2. Mercurial Vault

   - Repository: mercurial-dynamic-amm
   - Commit ID: 4e24a7fcb07c48fcc3d432ca1d417b15b6dadeff
   - Programs in scope:
     1. amm (programs/amm)

Out-of-scope: External libraries and financial related attacks.

2. ASSESSMENT SUMMARY & FINDINGS OVERVIEW

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL
0	0	1	4	1



EXECUTIVE OVERVIEW

SECURITY ANALYSIS	RISK LEVEL	REMEDIATION DATE
(HAL-01) - POTENTIAL DOS DUE TO INSECURE SEED FUNCTION	Medium	SOLVED - 07/21/2022
(HAL-02) - HARDCODED TREASURY AND BASE ADDRESSES	Low	FUTURE RELEASE
(HAL-03) - MISSING CARGO OVERFLOW CHECKS	Low	RISK ACCEPTED
(HAL-04) - MULTIPLE UNCHECKED ACCOUNTS	Low	RISK ACCEPTED
(HAL-05) - POSSIBLE RUST PANICS DUE TO UNSAFE UNWRAP USAGE	Informational	SOLVED - 08/02/2022
(HAL-06) - TIMESTAMP UNDERFLOW	Informational	ACKNOWLEDGED

FINDINGS & TECH DETAILS

3.1 (HAL-01) POTENTIAL DOS DUE TO INSECURE SEED FUNCTION - MEDIUM

Program Derived Addresses (PDAs) are Solana addresses that can be deterministically derived from a collection of seeds and a program ID. They are based on the ed25519 elliptic curve so same seeds derives same PDAs. In this case, Mercurial Finance is using an auxiliary function called form/_composite/_key which, trying to simplify developer's work, performs a bitwise OR among two seed address as follows:

Listing 1: dynamic-amm/programs/amm/src/utils.rs (Line 27)

```
22 /// Generate a public key from 2 keys by applying OR operator on
↳ the keys. The public key derived from the 2 keys is not affected
↳ by the key order.
23 pub fn form_composite_key(key1: [u8; 32], key2: [u8; 32]) -> [u8;
↳ 32] {
24     let mut symmetric_key: [u8; 32] = [0; 32];
25
26     for i in 0..key1.len() {
27         symmetric_key[i] = key1[i] | key2[i];
28     }
29
30     symmetric_key
31 }
```

The function above is used at the time of deriving a Pool PDA as follows:

Listing 2: dynamic-amm/programs/amm/src/context.rs (Line 21)

```
15 ...
16 pub struct Initialize<'info> {
17     #[account(
18         init,
19         seeds = [
20             constants::seeds::POOL_PREFIX.as_bytes(),
21             &form_composite_key(token_a_mint.key().to_bytes(),
↳ token_b_mint.key().to_bytes()),
22             &[curve_type.try_to_vec()?[0]],
23             base.key().as_ref()
24         ],
25         bump,
26         payer = admin,
27         // 8 - discriminator + 887 - pool + 1024 - extra byte
28         // Rent-exempt minimum: 0.010962 SOL
29         space = 8 + 887 + 1024 // This is safe, overflow won't
↳ happen
30     )]
31     ...
```

As it can be seen in the code, this function's intention is to avoid mistakes at the time of adding token mint seeds, since find_program_address is sensitive to the seed order, but when using the same base address and the same Curve type, it could cause address collisions because it does a bitwise OR to generate the mint seed.

This could prevent administrators from creating Pools and can be considered as a DoS (Denial-of-Service) vulnerability type.

For example, the following addresses, combined as mint1 + mint2 and mint1 + mint3, can generate the same pool address PDA:

- Mint Address 1: 4BNA3EoHBJ96vVCobD4ji7dyEnw9h4qdvPraiujVH48b
- Mint address 2: 4BNA3EoHBJ96vVCobD4ji7dyEnw9h4qdvPraiujVH48c
- Mint address 3: 4BNA3EoHBJ96vVCobD4ji7dyEnw9h4qdvPraiujVH48Y

The following PoC demonstrates this vulnerability:

Listing 3: Pool PDA collision PoC

```
pub async fn test_derived_pool_address(){

    let curve_type = CurveType::Stable { amp: 30 };
    let pool_base: Pubkey = Pubkey::new_unique();

    /* The following addresess are used to simplify the PoC,
↳ others could generate same results */
    let mint_address_1: Pubkey = Pubkey::from_str("4
↳ BNA3EoHBJ96vVCobD4ji7dyEnw9h4qdvPraiujVH48b").unwrap();
    let mint_address_2: Pubkey = Pubkey::from_str("4
↳ BNA3EoHBJ96vVCobD4ji7dyEnw9h4qdvPraiujVH48c").unwrap();
    let mint_address_3: Pubkey = Pubkey::from_str("4
↳ BNA3EoHBJ96vVCobD4ji7dyEnw9h4qdvPraiujVH48Y").unwrap();

    let derived_pubkey = Pubkey::find_program_address(
        &[b"pool".as_ref(),
            &amm::utils::form_composite_key(mint_address_1.
↳ to_bytes(), mint_address_2.to_bytes()),
            &[curve_type.try_to_vec().unwrap()[0]],
            pool_base.as_ref(),
        ],
        &amm::id(),
    );

    let derived_pubkey2 = Pubkey::find_program_address(
        &[b"pool".as_ref(),
            &amm::utils::form_composite_key(mint_address_1.
↳ to_bytes(), mint_address_3.to_bytes()),
            &[curve_type.try_to_vec().unwrap()[0]],
            pool_base.as_ref(),
        ],
        &amm::id(),
    );

    assert_ne!(derived_pubkey, derived_pubkey2);

}
```

Risk Level:

Likelihood - 5
Impact - 1

Recommendation:

We recommend avoiding using functions or hashing algorithms that can lead to collisions. For example, Anchor can randomize the pool address and delegating it to custom functions is not necessary.

References:

- Rust Find_program_address docs

Remediation Plan:

SOLVED: The Mercurial Finance team fixed this issue in commit 064b8950f6b49b2ad3a65e2e90e95c84d4b. Pool account addresses are now created using Anchor.

3.2 (HAL-02) HARDCODED TREASURY AND BASE ADDRESSES - LOW

Important treasury and base account addresses are hardcoded in mercurial-vault/programs/vault/src/libs.rs. In case those accounts are compromised or retired, the program authority has to redeploy the program to update them, increasing tech debt and operations cost.

Listing 4: mercurial-vault/programs/vault/src/libs.rs (Lines 42,47)

```
40 /// Treasury address
41 pub fn get_treasury_address() -> Pubkey {
42     Pubkey::from_str("9kZeN47U2dubGbbzMrzzoRAUvpuxVLRcjW9XiFpYjUo4
↳ ")
43         .expect("Must be correct Solana address")
44 }
45 /// Base address, setup by Mercurial
46 pub fn get_base_address() -> Pubkey {
47     Pubkey::from_str("HWzXGcGHy4tcpYfaRDCyLNzXqBTv3E6BttpCH2vJxArv
↳ ")
48         .expect("Must be correct Solana address")
49 }
```

Risk Level:

Likelihood - 2
Impact - 3

Recommendation:

Consider making those treasury and base addresses mutable and implement a function to update them in case they are compromised.

Remediation Plan:

PENDING: The Mercurial Finance team has specified that will change this functionality as soon as it has implemented governance functions.

3.3 (HAL-03) MISSING CARGO OVERFLOW CHECKS - LOW

Description:

It was observed that there is no overflow-checks=true in Cargo.toml. By default, overflow checks are disabled in optimized release builds. Hence, if there is an overflow in release builds, it will be silenced, leading to unexpected behavior of an application. Even if checked arithmetic is used through checked_*, it is recommended to have that check in Cargo.toml.

Code Location:

- programs/Cargo.toml files

Risk Level:

Likelihood - 2
Impact - 2

Recommendation:

It is recommended to add overflow-checks=true under your release profiles in Cargo.toml files.

Remediation Plan:

RISK ACCEPTED: The Mercurial Finance team accepted the risk of this finding.

3.4 (HAL-04) MULTIPLE UNCHECKED ACCOUNTS - LOW

Description:

The use of UncheckedAccounts when developing with Anchor should be a signal to the developer to pay extra attention to and thoroughly check the provided accounts instead of relying on built-in Anchor security controls.

Those accounts, since they are not assigned Anchor-generated discriminators, can be passed as accounts of other types, possibly leaving the program open to a Account Confusion Attack.

This attack takes advantage of unchecked data type and leverages accounts of unexpected type to tamper with the logic flow of programs.

Code Location:

An example:

Listing 5: mercurial-vault/programs/vault/src/context.rs (Lines 150,165)

```
144 pub struct InitializeStrategy<'info> {
145     /// Vault account
146     #[account(mut, has_one = admin)]
147     pub vault: Box<Account<'info, Vault>>,
148 
149     /// CHECK: Strategy program
150     pub strategy_program: UncheckedAccount<'info>,
151 
152     /// Strategy account
153     #[account(
154         init,
155         seeds = [
156             vault.key().as_ref(), reserve.key().as_ref(), &[bumps.
 ↳ strategy_index]
157         ],
```

```
158         bump,
159         payer = admin,
160         space = 1000 // more space for further update
161     )]
162     pub strategy: Box<Account<'info, Strategy>>,
163 
164     /// CHECK: Reserve account
165     pub reserve: UncheckedAccount<'info>,
```

Risk Level:

Likelihood - 3
Impact - 1

Recommendation:

Although Halborn have not found any vulnerability related to these accounts and all the tested ones were controlled programmatically, it is recommended to use defined Anchor data types and be careful when using unchecked data structures.

References:

- Anchor Development Team Twitter
- Anchor UncheckedAccount Reference

Remediation Plan:

RISK ACCEPTED: The Mercurial Finance team accepted the risk of this finding. When vaults interact with third-party programs, they always call to the hardcoded addresses (Solend program/Port finance program, etc.) even if the admin sends an incorrect program account.

3.5 (HAL-05) POSSIBLE RUST PANICS DUE TO UNSAFE UNWRAP USAGE - LOW

Description:

The use of helper methods in Rust, such as `unwrap`, is allowed in dev and testing environment because those methods are supposed to throw an error (also known as `panic!`) when called on `Option::None` or a `Result` which is not `Ok`. However, keeping `unwrap` functions in production environment is considered bad practice because they may lead to program crashes, which are usually accompanied by insufficient or misleading error messages.

Code Location:

An example of this may be the following lines:

Listing 6: mercurial-dynamic-amm/programs/amm/src/lib.rs (Line 109)

```
100         let pool = &mut ctx.accounts.pool;
101         pool.lp_mint = ctx.accounts.lp_mint.key();
102         pool.token_a_mint = ctx.accounts.token_a_mint.key();
103         pool.token_b_mint = ctx.accounts.token_b_mint.key();
104         pool.a_vault = ctx.accounts.a_vault.key();
105         pool.b_vault = ctx.accounts.b_vault.key();
106         pool.a_vault_lp = ctx.accounts.a_vault_lp.key();
107         pool.b_vault_lp = ctx.accounts.b_vault_lp.key();
108         pool.base = ctx.accounts.base.key();
109         pool.pool_bump = *ctx.bumps.get("pool").unwrap();
110         pool.enabled = true;
111         pool.admin_token_a_fee = ctx.accounts.admin_token_a_fee.
 ↳ key();
112         pool.admin_token_b_fee = ctx.accounts.admin_token_b_fee.
 ↳ key
```

Listing 7: mercurial-dynamic-amm/programs/amm/src/lib.rs (Line 967)

```
956
957         // Latest virtual price might error, when liquidity fully
 ↳ pulled out from the pool
958         let latest_virtual_price = if latest_virtual_price == None
 ↳  {
959             // We return the latest virtual price snapshot
960             // As the time-window keep on increasing, the APY
 ↳ should keep decreasing as the time pass away
961             let virtual_price = pool
962                 .snapshot
963                 .last()
964                 .ok_or(PoolError::InsufficientSnapshot)?;
965             virtual_price.price
966         } else {
967             latest_virtual_price.unwrap()
968         };
969
970         let current_time = Clock::get()?.unix_timestamp;
```

There are multiple incidences on state.rs file. The following is just one of them:

Listing 8: mercurial-dynamic-amm/programs/amm/src/state.rs (Line 216)

```
206         pub fn save_virtual_price(&mut self, virtual_price: u64,
 ↳ current_timestamp: i64) -> Result<()> {
207             let snapshot = VirtualPrice {
208                 price: virtual_price,
209                 timestamp: current_timestamp,
210             };
211             let last_snapshot = self.snapshot.last();
212             if last_snapshot.is_none() {
213                 self.snapshot.insert(&snapshot);
214                 return Ok(());
215             }
216             let last_snapshot = last_snapshot.unwrap();
217             let time_diff = current_timestamp - last_snapshot.
 ↳ timestamp;
218             let elapsed_window = time_diff / constants::snapshot::
 ↳ SIX_HOURS;
219             if elapsed_window > 0 {
220                 let mut void_window = elapsed_window - 1;
```

Risk Level:

Likelihood - 3
Impact - 1

Recommendation:

It is recommended not to use the unwrap function in the production environment because its use causes panic! and may crash the contract without verbose error messages. Crashing the system will result in a loss of availability and, in some cases, even private information stored in the state. Some alternatives are possible, such as propagating the error with ? instead of unwrapping, or using the error-chain crate for errors.

Remediation Plan:

SOLVED: The Mercurial Finance team has fixed this vulnerability in commit 46510f7e582d77bae0f396ecaa363a71805f06ab by taking adventage of vipers crate to get the values of bumps and options.

3.6 (HAL-06) TIMESTAMP UNDERFLOW - INFORMATIONAL

Integer overflow/underflow occurs when an arithmetic operation attempts to create a numeric value that is outside the range that can be represented by a given number of bits, either greater than the maximum or less than the minimum representable value. Although integer overflows and underflows do not cause Rust to panic in the release mode, the consequences could be dire if the result of those operations is used in financial calculations.

Although this vulnerability can represent a risk when found in other locations, it was verified the affected function (save_virtual_price) is only used in lib.rs file and retrieves the current_timestamp argument from a secure function (get_current_timestamp_u64).

As a result, this vulnerability has been marked as informational but that doesn't mean that a potential risk may arise if the affected function is used in other places in future versions or last_snapshot is compromised.

In this case, timestamps are used to calculate a time delta as follows:

Code Location:

Listing 9: mercurial-dynamic-amm/programs/amm/src/state.rs (Line 217)

```
206         pub fn save_virtual_price(&mut self, virtual_price: u64,
 ↳ current_timestamp: i64) -> Result<()> {
207             let snapshot = VirtualPrice {
208                 price: virtual_price,
209                 timestamp: current_timestamp,
210             };
211             let last_snapshot = self.snapshot.last();
212             if last_snapshot.is_none() {
213                 self.snapshot.insert(&snapshot);
214                 return Ok(());
215             }
216             let last_snapshot = last_snapshot.unwrap();
217             let time_diff = current_timestamp - last_snapshot.
 ↳ timestamp;
```

```
218         let elapsed_window = time_diff / constants::snapshot::
 ↳ SIX_HOURS;
219         if elapsed_window > 0 {
220             let mut void_window = elapsed_window - 1;
221             while void_window > 0 {
222                 self.snapshot.insert(&VirtualPrice {
223                     price: 0,
224                     timestamp: current_timestamp,
225                 });
226                 void_window -= 1;
227             }
228             self.snapshot.insert(&snapshot);
229         }
230         Ok(())
231     }
```

Recommendation:

It is recommended to use safe and verified math libraries (such as checked_sub, checked_div) for consistent arithmetic operations throughout the Solana program system. Consider using Rust safe arithmetic functions for primitives instead of standard arithmetic operators.

Reference:

Safe arithmetic operations for primitives: u8, u32 & u64

Remediation Plan:

ACKNOWLEDGED: The Mercurial Finance team acknowledged this finding.

AUTOMATED TESTING

Description:

Halborn used automated security scanners to assist with detection of well-known security issues, and to identify low-hanging fruits on the targets for this engagement. Among the tools used was Soteria, a security analysis service for Solana programs. Soteria performed a scan on all the programs and sent the compiled results to the analyzers to locate any vulnerabilities.

Results:

Please see the following page.

AUTOMATED TESTING

```
anchor_lang_version: 0.24.2 anchorVersionTooOld: 0
 - ✔ [00m:00s] Loading IR From File
 - ▮ [00m:00s] Running Compiler Optimization Passes
EntryPoints:
entrypoint
 - ✔ [00m:00s] Running Compiler Optimization Passes
 - ✔ [00m:00s] Running Pointer Analysis
=============This arithmetic operation may be UNSAFE!================
Found a potential vulnerability at line 985, column 44 in programs/amm/src/lib.rs
The div operation may result in overflows:

 979|            .ok_or(PoolError::InsufficientSnapshot)?;
 980|        let first = utils::u64_to_float(
 981|            first_virtual_price.price,
 982|            constants::virtual_price::DECIMAL as i32,
 983|        )
 984|        .ok_or(PoolError::ConversionError)?;
>985|        let time_elapsed = f64::value_from(current_time - first_virtual_price.timestamp)
 986|            .map_err(|_| PoolError::ConversionError)?;
 987|        let apy = utils::get_apy(first, second, time_elapsed);
 988|        emit!(amm::event::PoolInfo {
 989|            token_a_amount,
 990|            token_b_amount,
 991|            virtual_price: second,
>>>Stack Trace:
>>>amm::try_entry::h345a39e05ededed4 [programs/amm/src/lib.rs:55]
>>>  amm::dispatch::h4dcbaf9ab19e038d [programs/amm/src/lib.rs:55]
>>>    amm::__private::__global::get_pool_info::h0cc786f4bb0c234b [programs/amm/src/lib.rs:55]
>>>      amm::amm::get_pool_info::h6ece67e261f15533 [programs/amm/src/lib.rs:55]

=============This arithmetic operation may be UNSAFE!================
Found a potential vulnerability at line 217, column 25 in programs/amm/src/state.rs
The div operation may result in overflows:

 211|        let last_snapshot = self.snapshot.last();
 212|        if last_snapshot.is_none() {
 213|            self.snapshot.insert(&snapshot);
 214|            return Ok(());
 215|        }
 216|        let last_snapshot = last_snapshot.unwrap();
>217|        let time_diff = current_timestamp - last_snapshot.timestamp;
 218|        let elapsed_window = time_diff / constants::snapshot::SIX_HOURS;
 219|        if elapsed_window > 0 {
 220|            let mut void_window = elapsed_window - 1;
 221|            while void_window > 0 {
 222|                self.snapshot.insert(&VirtualPrice {
 223|                    price: 0,
>>>Stack Trace:
>>>amm::try_entry::h345a39e05ededed4 [programs/amm/src/lib.rs:55]
>>>  amm::dispatch::h4dcbaf9ab19e038d [programs/amm/src/lib.rs:55]
>>>    amm::__private::__global::sync_apy::hba4538afb47db645 [programs/amm/src/lib.rs:55]
>>>      amm::amm::sync_apy::h5073d52b4d978918 [programs/amm/src/lib.rs:55]
>>>        amm::state::Pool::save_virtual_price::h002cbb0c39e0e91e [programs/amm/src/lib.rs:930]
```

4.1 AUTOMATED ANALYSIS

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.

Results:

ID	package	Short Description
RUSTSEC-2020-0159	chrono	Potential segfault in ‘localtime_r‘ invocations
RUSTSEC-2020-0071	time	Potential segfault in the time crate

4.2 UNSAFE RUST CODE DETECTION

Description:

Halborn used automated security scanners to assist with the detection of well-known security issues and vulnerabilities. Among the tools used was cargo-geiger, a security tool that lists statistics related to the usage of unsafe Rust code in a core Rust codebase and all its dependencies.

AUTOMATED TESTING

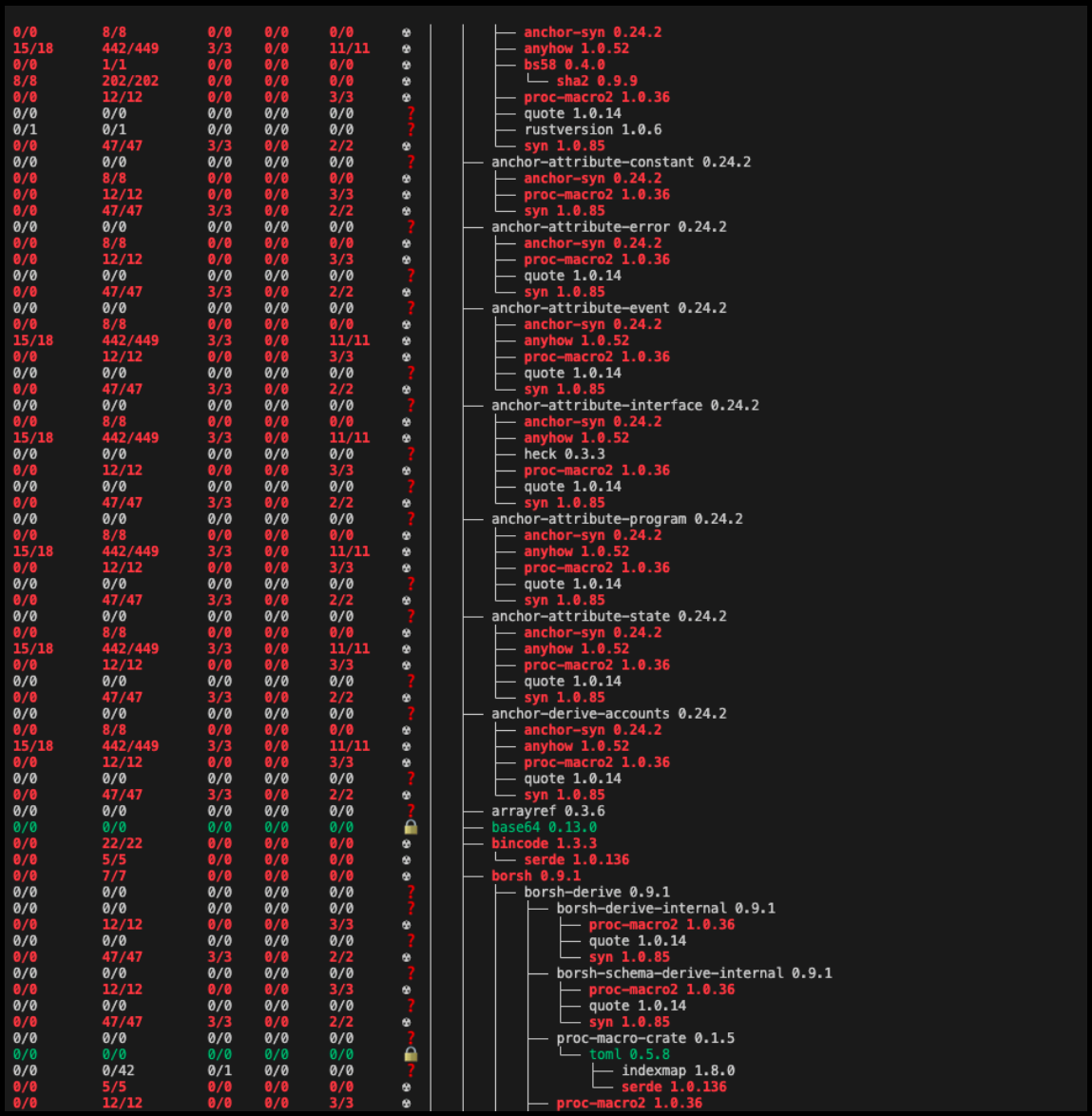
AUTOMATED TESTING

Vault Results:

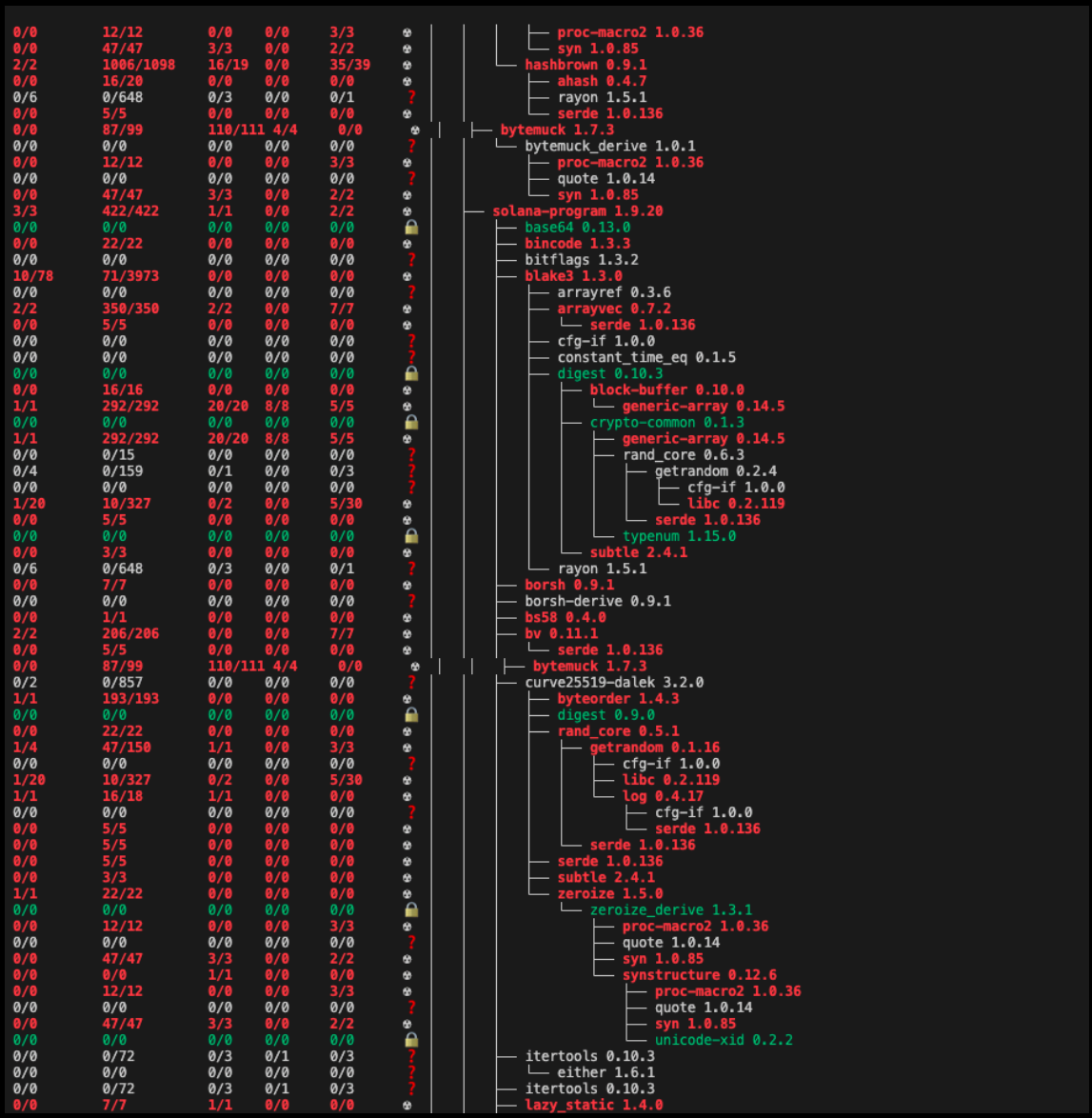
```
Metric output format: x/y
    x = unsafe code used by the build
    y = total unsafe code found in the crate

Symbols:
    🔒  = No `unsafe` usage found, declares #![forbid(unsafe_code)]
    ?   = No `unsafe` usage found, missing #![forbid(unsafe_code)]
    ☢   = `unsafe` usage found

Functions  Expressions  Impls  Traits  Methods  Dependency

0/0        0/0          0/0    0/0     0/0      ?  mercurial-vault 0.5.0
0/0        0/0          0/0    0/0     0/0      ?  ├── anchor-lang 0.24.2
0/0        0/0          0/0    0/0     0/0      ?  │   ├── anchor-attribute-access-control 0.24.2
0/0        8/8          0/0    0/0     0/0      ☢  │   │   ├── anchor-syn 0.24.2
15/18      442/449      3/3    0/0     11/11    ☢  │   │   │   ├── anyhow 1.0.52
0/22       0/775        0/6    0/0     0/5      ?  │   │   │   │   └── backtrace 0.3.63
0/0        0/26         0/0    0/0     0/0      ?  │   │   │   │       ├── addr2line 0.17.0
0/0        0/48         0/3    0/1     0/0      ?  │   │   │   │       │   ├── gimli 0.26.1
0/0        0/42         0/1    0/0     0/0      ?  │   │   │   │       │   │   ├── indexmap 1.8.0
0/2        0/1198       0/22   0/1     0/58     ?  │   │   │   │       │   │   │   ├── hashbrown 0.11.2
4/6        389/1102     3/9    1/1     12/25    ☢  │   │   │   │       │   │   │   │   ├── bumpalo 3.9.1
0/6        0/648        0/3    0/0     0/1      ?  │   │   │   │       │   │   │   │   ├── rayon 1.5.1
0/0        0/451        0/6    0/0     0/6      ?  │   │   │   │       │   │   │   │   │   ├── crossbeam-deque 0.8.1
0/0        0/0          0/0    0/0     0/0      ?  │   │   │   │       │   │   │   │   │   │   ├── cfg-if 1.0.0
0/3        0/433        0/9    0/0     0/26     ?  │   │   │   │       │   │   │   │   │   │   ├── crossbeam-epoch 0.9.6
0/0        0/0          0/0    0/0     0/0      ?  │   │   │   │       │   │   │   │   │   │   │   ├── cfg-if 1.0.0
0/4        0/79         0/14   0/0     0/2      ?  │   │   │   │       │   │   │   │   │   │   │   ├── crossbeam-utils 0.8.6
0/0        0/0          0/0    0/0     0/0      ?  │   │   │   │       │   │   │   │   │   │   │   │   ├── cfg-if 1.0.0
0/0        7/7          1/1    0/0     0/0      ☢  │   │   │   │       │   │   │   │   │   │   │   │   └── lazy_static 1.4.0
0/0        0/49         0/6    0/0     0/3      ?  │   │   │   │       │   │   │   │   │   │   │   │       └── spin 0.5.2
0/0        7/7          1/1    0/0     0/0      ☢  │   │   │   │       │   │   │   │   │   │   │   ├── lazy_static 1.4.0
0/0        0/0          0/0    0/0     0/0      ?  │   │   │   │       │   │   │   │   │   │   │   ├── memoffset 0.6.5
0/0        0/18         0/1    0/0     0/0      ?  │   │   │   │       │   │   │   │   │   │   │   └── scopeguard 1.1.0
0/4        0/79         0/14   0/0     0/2      ?  │   │   │   │       │   │   │   │   │   │   └── crossbeam-utils 0.8.6
0/0        0/0          0/0    0/0     0/0      ?  │   │   │   │       │   │   │   │   │   ├── either 1.6.1
0/0        5/5          0/0    0/0     0/0      ☢  │   │   │   │       │   │   │   │   │   │   └── serde 1.0.136
0/0        0/0          0/0    0/0     0/0      ?  │   │   │   │       │   │   │   │   │   │       └── serde_derive 1.0.136
0/0        12/12        0/0    0/0     3/3      ☢  │   │   │   │       │   │   │   │   │   │           ├── proc-macro2 1.0.36
0/0        0/0          0/0    0/0     0/0      🔒 │   │   │   │       │   │   │   │   │   │           │   └── unicode-xid 0.2.2
0/0        0/0          0/0    0/0     0/0      ?  │   │   │   │       │   │   │   │   │   │           ├── quote 1.0.14
0/0        12/12        0/0    0/0     3/3      ☢  │   │   │   │       │   │   │   │   │   │           │   └── proc-macro2 1.0.36
0/0        47/47        3/3    0/0     2/2      ☢  │   │   │   │       │   │   │   │   │   │           └── syn 1.0.85
0/0        12/12        0/0    0/0     3/3      ☢  │   │   │   │       │   │   │   │   │   │               ├── proc-macro2 1.0.36
0/0        0/0          0/0    0/0     0/0      ?  │   │   │   │       │   │   │   │   │   │               ├── quote 1.0.14
0/0        0/0          0/0    0/0     0/0      🔒 │   │   │   │       │   │   │   │   │   │               └── unicode-xid 0.2.2
0/5        0/488        0/2    0/0     0/20     ?  │   │   │   │       │   │   │   │   │   └── rayon-core 1.9.1
0/2        0/518        0/7    0/0     0/14     ?  │   │   │   │       │   │   │   │   │       ├── crossbeam-channel 0.5.2
0/0        0/0          0/0    0/0     0/0      ?  │   │   │   │       │   │   │   │   │       │   ├── cfg-if 1.0.0
0/4        0/79         0/14   0/0     0/2      ?  │   │   │   │       │   │   │   │   │       │   └── crossbeam-utils 0.8.6
0/0        0/451        0/6    0/0     0/6      ?  │   │   │   │       │   │   │   │   │       ├── crossbeam-deque 0.8.1
0/4        0/79         0/14   0/0     0/2      ?  │   │   │   │       │   │   │   │   │       ├── crossbeam-utils 0.8.6
0/0        7/7          1/1    0/0     0/0      ☢  │   │   │   │       │   │   │   │   │       ├── lazy_static 1.4.0
0/0        0/72         0/0    0/0     0/0      ?  │   │   │   │       │   │   │   │   │       └── num_cpus 1.13.1
1/20       10/327       0/2    0/0     5/30     ☢  │   │   │   │       │   │   │   │   │           └── libc 0.2.119
0/0        5/5          0/0    0/0     0/0      ☢  │   │   │   │       │   │   │   │   └── serde 1.0.136
0/6        0/648        0/3    0/0     0/1      ?  │   │   │   │       │   │   │   ├── rayon 1.5.1
0/0        5/5          0/0    0/0     0/0      ☢  │   │   │   │       │   │   │   └── serde 1.0.136
0/0        0/0          0/18   0/2     0/0      ?  │   │   │   │       │   │   └── stable_deref_trait 1.2.0
0/0        0/26         0/0    0/1     0/0      ?  │   │   │   │       │   ├── object 0.27.1
0/6        0/156        0/0    0/0     0/0      ?  │   │   │   │       │   │   ├── crc32fast 1.3.0
0/0        0/0          0/0    0/0     0/0      ?  │   │   │   │       │   │   │   └── cfg-if 1.0.0
0/2        0/120        0/2    0/0     0/2      ?  │   │   │   │       │   │   ├── flate2 1.0.22
0/0        0/0          0/0    0/0     0/0      ?  │   │   │   │       │   │   │   ├── cfg-if 1.0.0
0/6        0/156        0/0    0/0     0/0      ?  │   │   │   │       │   │   │   ├── crc32fast 1.3.0
1/20       10/327       0/2    0/0     5/30     ☢  │   │   │   │       │   │   │   ├── libc 0.2.119
0/0        0/0          0/0    0/0     0/0      🔒 │   │   │   │       │   │   │   └── miniz_oxide 0.4.4
```

```
0/0        0/0          0/0    0/0     0/0      🔒 │   │   │   │       │   │   │       └── miniz_oxide 0.4.4
0/0        0/0          0/0    0/0     0/0      🔒 │   │   │   │       │   │   │           └── adler 1.0.2
0/0        0/42         0/1    0/0     0/0      ?  │   │   │   │       │   │   ├── indexmap 1.8.0
36/37      2067/2140    0/0    0/0     16/16    ☢  │   │   │   │       │   │   └── memchr 2.4.1
1/20       10/327       0/2    0/0     5/30     ☢  │   │   │   │       │   │       └── libc 0.2.119
0/0        0/0          0/0    0/0     0/0      ?  │   │   │   │       │   ├── rustc-demangle 0.1.21
0/1        0/392        0/7    0/1     0/13     ?  │   │   │   │       │   └── smallvec 1.8.0
0/0        5/5          0/0    0/0     0/0      ☢  │   │   │   │       │       └── serde 1.0.136
0/0        0/0          0/0    0/0     0/0      ?  │   │   │   │       ├── cfg-if 1.0.0
1/20       10/327       0/2    0/0     5/30     ☢  │   │   │   │       ├── libc 0.2.119
0/0        0/0          0/0    0/0     0/0      🔒 │   │   │   │       ├── miniz_oxide 0.4.4
0/0        0/26         0/0    0/1     0/0      ?  │   │   │   │       ├── object 0.27.1
0/0        0/0          0/0    0/0     0/0      ?  │   │   │   │       ├── rustc-demangle 0.1.21
0/0        5/5          0/0    0/0     0/0      ☢  │   │   │   │       └── serde 1.0.136
0/0        1/1          0/0    0/0     0/0      ☢  │   │   ├── bs58 0.3.1
0/0        0/0          0/0    0/0     0/0      ?  │   │   ├── heck 0.3.3
0/0        0/0          0/0    0/0     0/0      ?  │   │   │   └── unicode-segmentation 1.8.0
0/0        12/12        0/0    0/0     3/3      ☢  │   │   ├── proc-macro2 1.0.36
0/0        0/0          0/0    0/0     0/0      ?  │   │   ├── proc-macro2-diagnostics 0.9.1
0/0        12/12        0/0    0/0     3/3      ☢  │   │   │   ├── proc-macro2 1.0.36
0/0        0/0          0/0    0/0     0/0      ?  │   │   │   ├── quote 1.0.14
0/0        47/47        3/3    0/0     2/2      ☢  │   │   │   ├── syn 1.0.85
3/3        34/34        0/0    0/0     0/0      ☢  │   │   │   └── yansi 0.5.0
0/0        0/0          0/0    0/0     0/0      ?  │   │   ├── quote 1.0.14
0/0        5/5          0/0    0/0     0/0      ☢  │   │   ├── serde 1.0.136
0/0        4/7          0/0    0/0     0/0      ☢  │   │   ├── serde_json 1.0.75
0/0        0/42         0/1    0/0     0/0      ?  │   │   │   ├── indexmap 1.8.0
0/0        7/7          0/0    0/0     0/0      ☢  │   │   │   ├── itoa 1.0.1
7/9        587/723      0/0    0/0     2/2      ☢  │   │   │   ├── ryu 1.0.9
0/0        5/5          0/0    0/0     0/0      ☢  │   │   │   └── serde 1.0.136
8/8        202/202      0/0    0/0     0/0      ☢  │   │   ├── sha2 0.9.9
0/0        6/6          0/0    0/0     0/0      ☢  │   │   │   ├── block-buffer 0.9.0
0/0        3/3          0/0    0/0     0/0      ☢  │   │   │   │   ├── block-padding 0.2.1
1/1        292/292      20/20  8/8     5/5      ☢  │   │   │   │   └── generic-array 0.14.5
0/0        5/5          0/0    0/0     0/0      ☢  │   │   │   │       ├── serde 1.0.136
0/0        0/0          0/0    0/0     0/0      🔒 │   │   │   │       └── typenum 1.15.0
0/0        0/0          0/0    0/0     0/0      ?  │   │   │   ├── cfg-if 1.0.0
0/1        0/14         0/0    0/0     0/0      ?  │   │   │   ├── cpufeatures 0.2.1
0/0        0/0          0/0    0/0     0/0      🔒 │   │   │   ├── digest 0.9.0
```

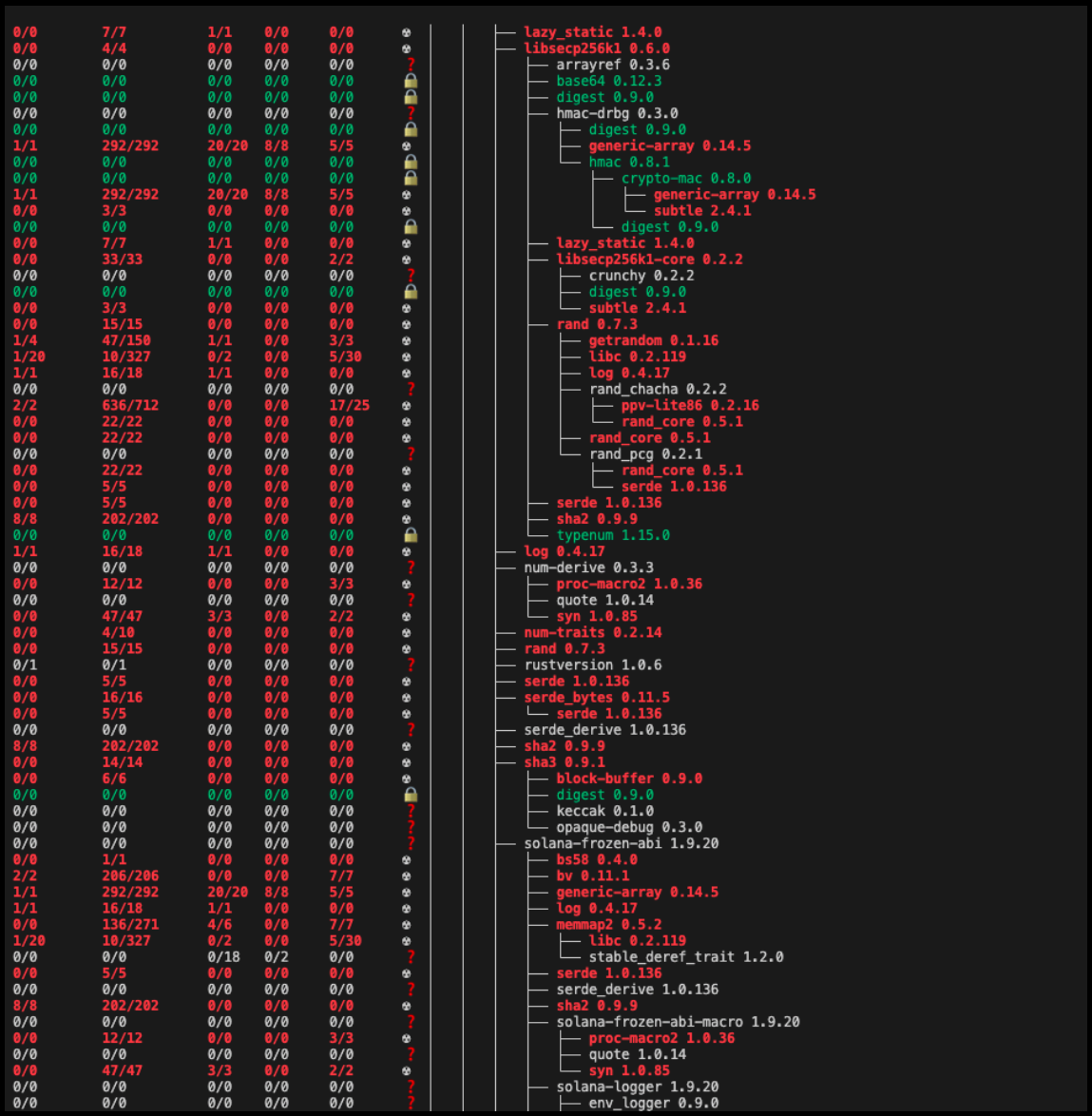
AUTOMATED TESTING



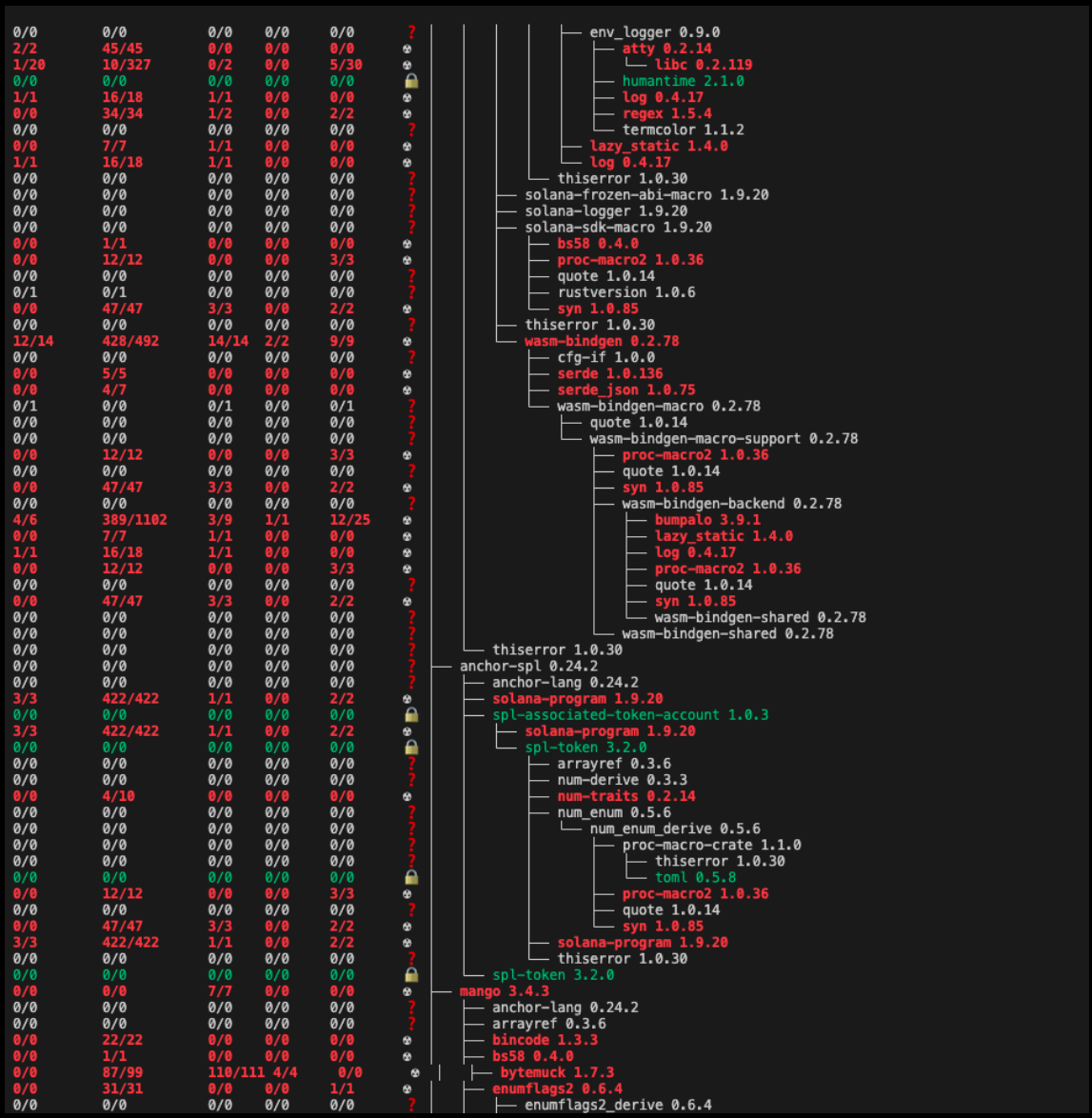
AUTOMATED TESTING



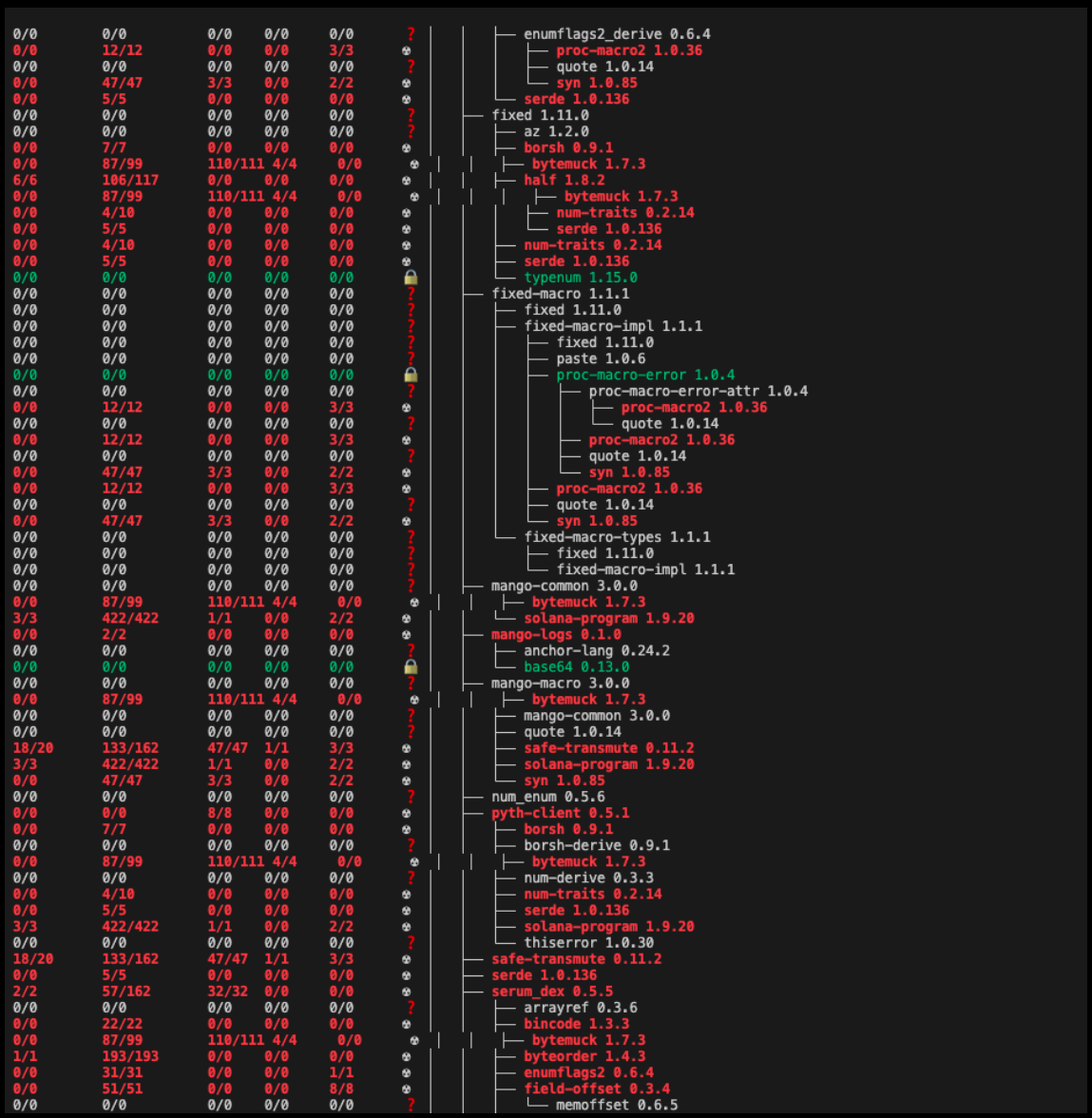
AUTOMATED TESTING



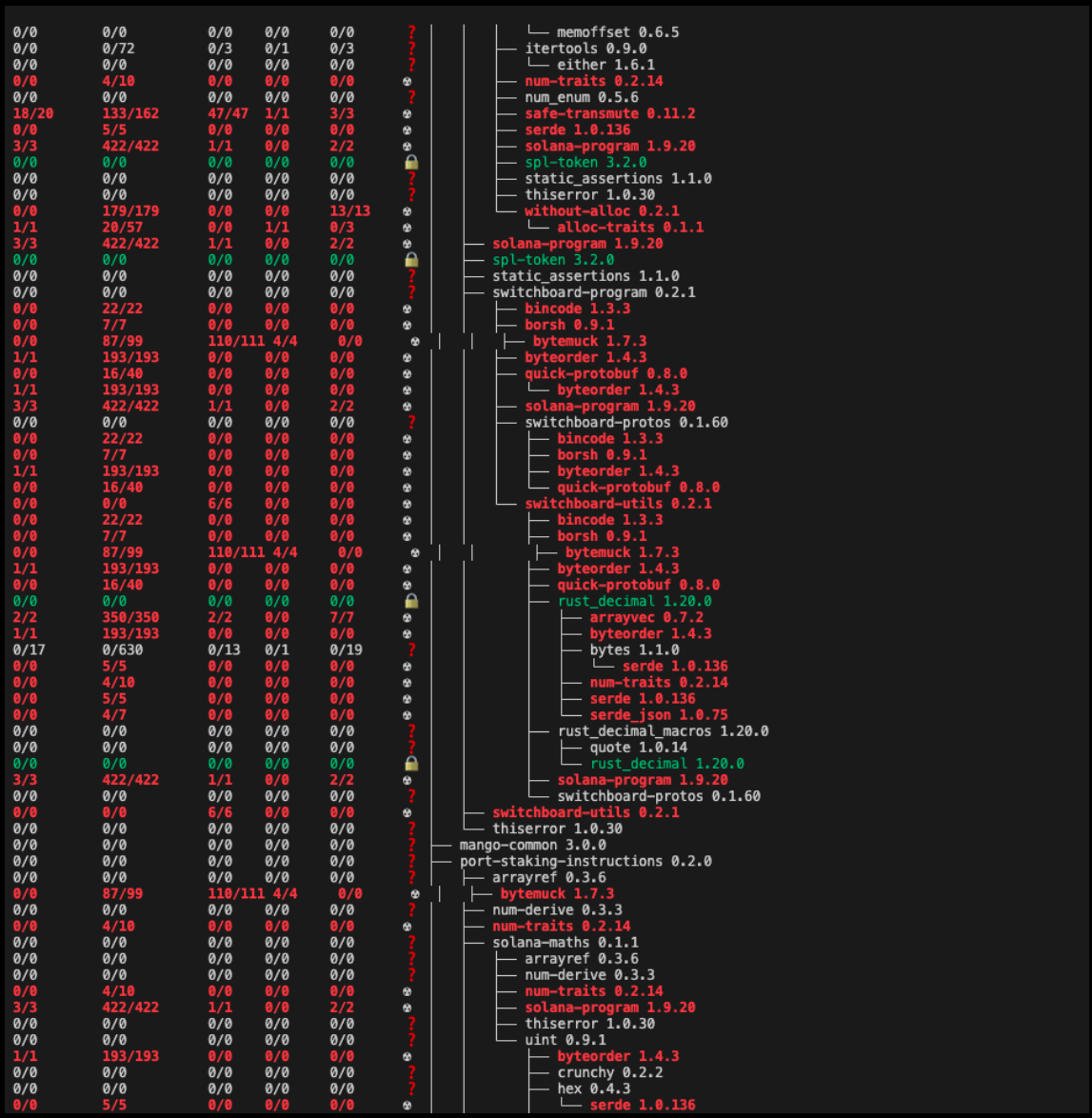
AUTOMATED TESTING

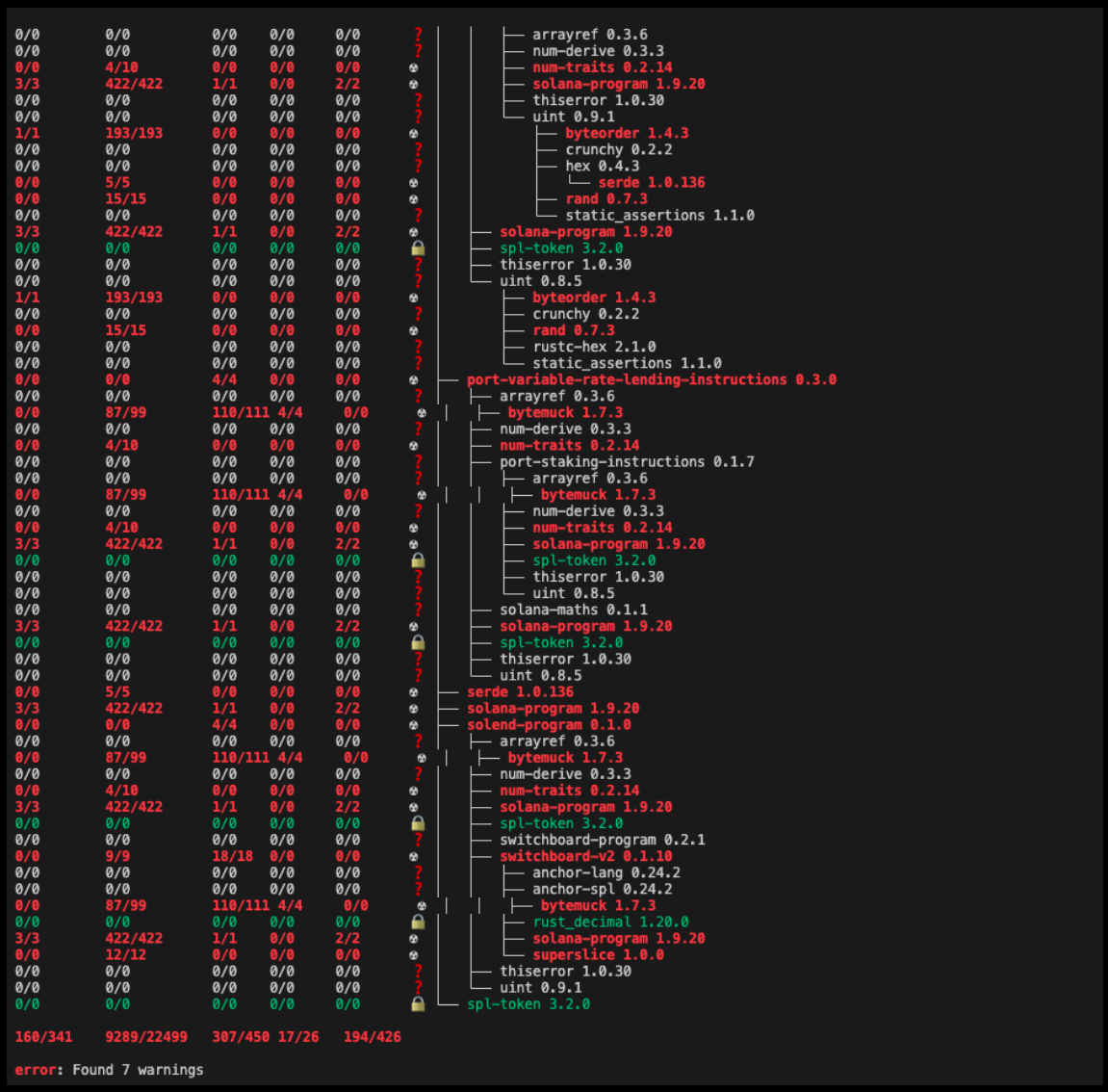


AUTOMATED TESTING



AUTOMATED TESTING




0/0 0/0 0/0 0/0 0/0 ? ├── arrayref 0.3.6
0/0 0/0 0/0 0/0 0/0 ? ├── num-derive 0.3.3
0/0 4/10 0/0 0/0 0/0 ☢ ├── num-traits 0.2.14
3/3 422/422 1/1 0/0 2/2 ☢ ├── solana-program 1.9.20
0/0 0/0 0/0 0/0 0/0 ? ├── thiserror 1.0.30
0/0 0/0 0/0 0/0 0/0 ? └── uint 0.9.1
1/1 193/193 0/0 0/0 0/0 ☢ ├── byteorder 1.4.3
0/0 0/0 0/0 0/0 0/0 ? ├── crunchy 0.2.2
0/0 0/0 0/0 0/0 0/0 ? ├── hex 0.4.3
0/0 5/5 0/0 0/0 0/0 ☢ │ └── serde 1.0.136
0/0 15/15 0/0 0/0 0/0 ☢ ├── rand 0.7.3
0/0 0/0 0/0 0/0 0/0 ? └── static_assertions 1.1.0
3/3 422/422 1/1 0/0 2/2 ☢ ├── solana-program 1.9.20
0/0 0/0 0/0 0/0 0/0 🔒 ├── spl-token 3.2.0
0/0 0/0 0/0 0/0 0/0 ? ├── thiserror 1.0.30
0/0 0/0 0/0 0/0 0/0 ? └── uint 0.8.5
1/1 193/193 0/0 0/0 0/0 ☢ ├── byteorder 1.4.3
0/0 0/0 0/0 0/0 0/0 ? ├── crunchy 0.2.2
0/0 15/15 0/0 0/0 0/0 ☢ ├── rand 0.7.3
0/0 0/0 0/0 0/0 0/0 ? ├── rustc-hex 2.1.0
0/0 0/0 0/0 0/0 0/0 ? └── static_assertions 1.1.0
0/0 0/0 4/4 0/0 0/0 ☢ ├── port-variable-rate-lending-instructions 0.3.0
0/0 0/0 0/0 0/0 0/0 ? ├── arrayref 0.3.6
0/0 87/99 110/111 4/4 0/0 ☢ ├── bytemuck 1.7.3
0/0 0/0 0/0 0/0 0/0 ? ├── num-derive 0.3.3
0/0 4/10 0/0 0/0 0/0 ☢ ├── num-traits 0.2.14
0/0 0/0 0/0 0/0 0/0 ? ├── port-staking-instructions 0.1.7
0/0 0/0 0/0 0/0 0/0 ? ├── arrayref 0.3.6
0/0 87/99 110/111 4/4 0/0 ☢ ├── bytemuck 1.7.3
0/0 0/0 0/0 0/0 0/0 ? ├── num-derive 0.3.3
0/0 4/10 0/0 0/0 0/0 ☢ ├── num-traits 0.2.14
3/3 422/422 1/1 0/0 2/2 ☢ ├── solana-program 1.9.20
0/0 0/0 0/0 0/0 0/0 🔒 ├── spl-token 3.2.0
0/0 0/0 0/0 0/0 0/0 ? ├── thiserror 1.0.30
0/0 0/0 0/0 0/0 0/0 ? └── uint 0.8.5
0/0 0/0 0/0 0/0 0/0 ? ├── solana-maths 0.1.1
3/3 422/422 1/1 0/0 2/2 ☢ ├── solana-program 1.9.20
0/0 0/0 0/0 0/0 0/0 🔒 ├── spl-token 3.2.0
0/0 0/0 0/0 0/0 0/0 ? ├── thiserror 1.0.30
0/0 0/0 0/0 0/0 0/0 ? └── uint 0.8.5
0/0 5/5 0/0 0/0 0/0 ☢ ├── serde 1.0.136
3/3 422/422 1/1 0/0 2/2 ☢ ├── solana-program 1.9.20
0/0 0/0 4/4 0/0 0/0 ☢ ├── solend-program 0.1.0
0/0 0/0 0/0 0/0 0/0 ? ├── arrayref 0.3.6
0/0 87/99 110/111 4/4 0/0 ☢ ├── bytemuck 1.7.3
0/0 0/0 0/0 0/0 0/0 ? ├── num-derive 0.3.3
0/0 4/10 0/0 0/0 0/0 ☢ ├── num-traits 0.2.14
3/3 422/422 1/1 0/0 2/2 ☢ ├── solana-program 1.9.20
0/0 0/0 0/0 0/0 0/0 🔒 ├── spl-token 3.2.0
0/0 0/0 0/0 0/0 0/0 ? ├── switchboard-program 0.2.1
0/0 9/9 18/18 0/0 0/0 ☢ ├── switchboard-v2 0.1.10
0/0 0/0 0/0 0/0 0/0 ? ├── anchor-lang 0.24.2
0/0 0/0 0/0 0/0 0/0 ? ├── anchor-spl 0.24.2
0/0 87/99 110/111 4/4 0/0 ☢ ├── bytemuck 1.7.3
0/0 0/0 0/0 0/0 0/0 🔒 ├── rust_decimal 1.20.0
3/3 422/422 1/1 0/0 2/2 ☢ ├── solana-program 1.9.20
0/0 12/12 0/0 0/0 0/0 ☢ └── superslice 1.0.0
0/0 0/0 0/0 0/0 0/0 ? ├── thiserror 1.0.30
0/0 0/0 0/0 0/0 0/0 ? └── uint 0.9.1
0/0 0/0 0/0 0/0 0/0 🔒 └── spl-token 3.2.0
160/341 9289/22499 307/450 17/26 194/426
error: Found 7 warnings

AUTOMATED TESTING

AUTOMATED TESTING

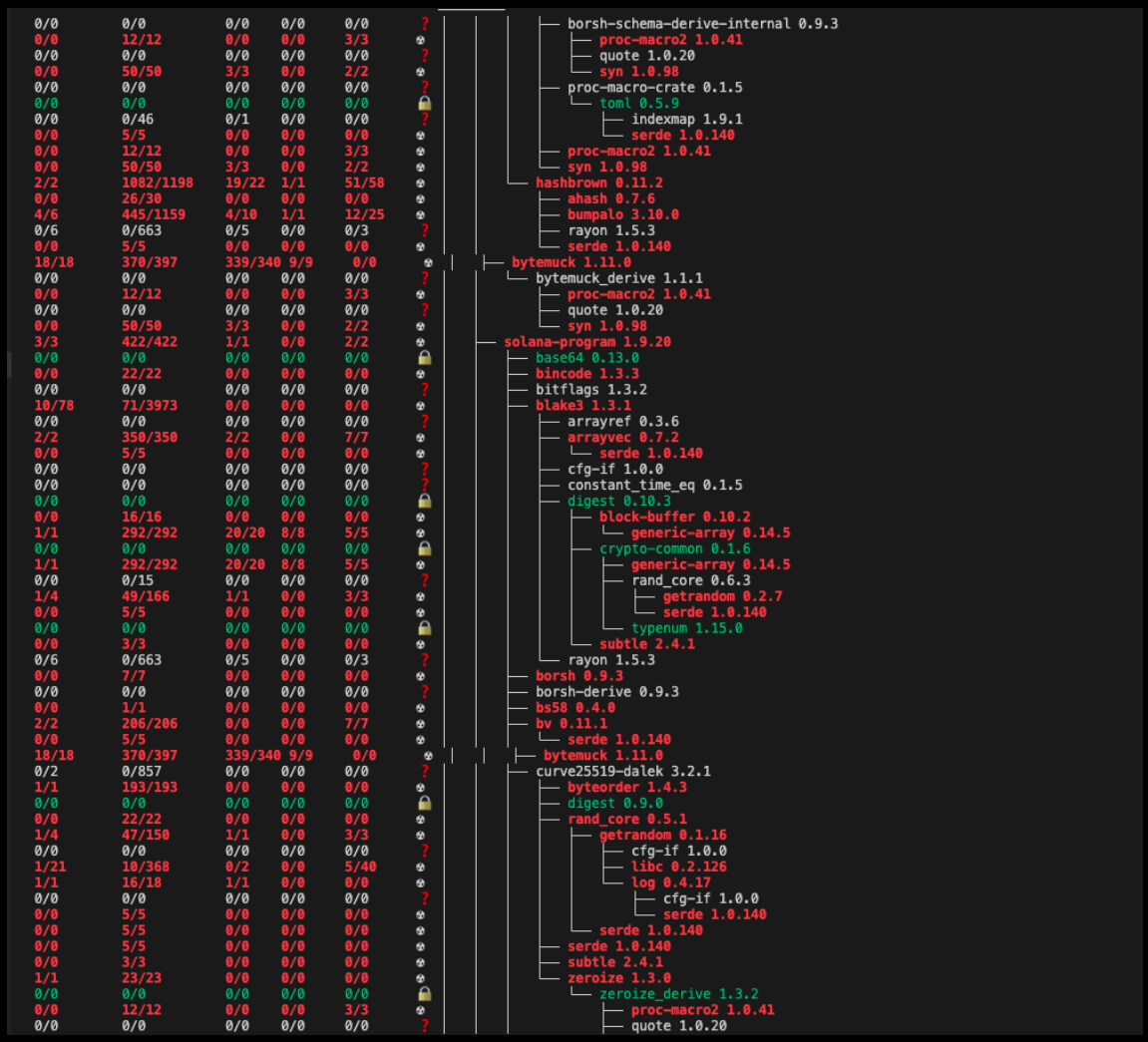
Dynamic AMM Results:

```
Metric output format: x/y
    x = unsafe code used by the build
    y = total unsafe code found in the crate

Symbols:
    🔒 = No `unsafe` usage found, declares #![forbid(unsafe_code)]
    ❓ = No `unsafe` usage found, missing #![forbid(unsafe_code)]
    ☢  = `unsafe` usage found

Functions  Expressions  Impls  Traits  Methods  Dependency

0/0        0/0          0/0    0/0     0/0      ❓ amm 0.2.1
0/0        0/0          0/0    0/0     0/0      ❓ ├── anchor-lang 0.24.2
0/0        0/0          0/0    0/0     0/0      ❓ │   ├── anchor-attribute-access-control 0.24.2
0/0        8/8          0/0    0/0     0/0      ☢  │   │   └── anchor-syn 0.24.2
15/18      442/449      3/3    0/0     11/11    ☢  │   │       ├── anyhow 1.0.58
0/22       0/791        0/6    0/0     0/5      ❓ │   │       │   └── backtrace 0.3.66
0/0        0/26         0/0    0/0     0/0      ❓ │   │       │       ├── addr2line 0.17.0
0/0        0/48         0/3    0/1     0/0      ❓ │   │       │       │   ├── gimli 0.26.2
0/0        0/46         0/1    0/0     0/0      ❓ │   │       │       │   │   └── indexmap 1.9.1
0/1        0/1367       0/24   0/1     0/69     ❓ │   │       │       │   │       ├── hashbrown 0.12.3
0/0        26/30        0/0    0/0     0/0      ☢  │   │       │       │   │       │   ├── ahash 0.7.6
1/4        49/166       1/1    0/0     3/3      ☢  │   │       │       │   │       │   │   ├── getrandom 0.2.7
0/0        0/0          0/0    0/0     0/0      ❓ │   │       │       │   │       │   │   │   ├── cfg-if 1.0.0
1/21       10/368       0/2    0/0     5/40     ☢  │   │       │       │   │       │   │   │   └── libc 0.2.126
1/1        75/117       4/6    0/0     2/3      ☢  │   │       │       │   │       │   │   ├── once_cell 1.13.0
0/16       0/1341       0/0    0/0     0/56     ❓ │   │       │       │   │       │   │   │   └── parking_lot_core 0.9.3
0/22       0/791        0/6    0/0     0/5      ❓ │   │       │       │   │       │   │   │       ├── backtrace 0.3.66
0/0        0/0          0/0    0/0     0/0      ❓ │   │       │       │   │       │   │   │       ├── cfg-if 1.0.0
1/21       10/368       0/2    0/0     5/40     ☢  │   │       │       │   │       │   │   │       ├── libc 0.2.126
0/1        0/399        0/7    0/1     0/13     ❓ │   │       │       │   │       │   │   │       └── smallvec 1.9.0
0/0        5/5          0/0    0/0     0/0      ☢  │   │       │       │   │       │   │   │           └── serde 1.0.140
0/0        0/0          0/0    0/0     0/0      ❓ │   │       │       │   │       │   │   │               └── serde_derive 1.0.140
0/0        12/12        0/0    0/0     3/3      ☢  │   │       │       │   │       │   │   │                   ├── proc-macro2 1.0.41
0/0        4/4          0/0    0/0     0/0      ☢  │   │       │       │   │       │   │   │                   │   └── unicode-ident 1.0.2
0/0        0/0          0/0    0/0     0/0      ❓ │   │       │       │   │       │   │   │                   ├── quote 1.0.20
0/0        12/12        0/0    0/0     3/3      ☢  │   │       │       │   │       │   │   │                   │   └── proc-macro2 1.0.41
0/0        50/50        3/3    0/0     2/2      ☢  │   │       │       │   │       │   │   │                   └── syn 1.0.98
0/0        12/12        0/0    0/0     3/3      ☢  │   │       │       │   │       │   │   │                       ├── proc-macro2 1.0.41
0/0        0/0          0/0    0/0     0/0      ❓ │   │       │       │   │       │   │   │                       ├── quote 1.0.20
0/0        4/4          0/0    0/0     0/0      ☢  │   │       │       │   │       │   │   │                       └── unicode-ident 1.0.2
0/0        5/5          0/0    0/0     0/0      ☢  │   │       │       │   │       │   │   └── serde 1.0.140
4/6        445/1159     4/10   1/1     12/25    ☢  │   │       │       │   │       │   ├── bumpalo 3.10.0
0/6        0/663        0/5    0/0     0/3      ❓ │   │       │       │   │       │   ├── rayon 1.5.3
0/0        0/453        0/6    0/0     0/6      ❓ │   │       │       │   │       │   │   ├── crossbeam-deque 0.8.2
0/0        0/0          0/0    0/0     0/0      ❓ │   │       │       │   │       │   │   │   ├── cfg-if 1.0.0
0/3        0/447        0/9    0/0     0/28     ❓ │   │       │       │   │       │   │   │   ├── crossbeam-epoch 0.9.10
0/0        0/0          0/0    0/0     0/0      ❓ │   │       │       │   │       │   │   │   │   ├── cfg-if 1.0.0
0/4        0/81         0/14   0/0     0/2      ❓ │   │       │       │   │       │   │   │   │   ├── crossbeam-utils 0.8.11
0/0        0/0          0/0    0/0     0/0      ❓ │   │       │       │   │       │   │   │   │   │   ├── cfg-if 1.0.0
1/1        75/117       4/6    0/0     2/3      ☢  │   │       │       │   │       │   │   │   │   │   └── once_cell 1.13.0
0/0        0/0          0/0    0/0     0/0      ❓ │   │       │       │   │       │   │   │   │   ├── memoffset 0.6.5
1/1        75/117       4/6    0/0     2/3      ☢  │   │       │       │   │       │   │   │   │   ├── once_cell 1.13.0
0/0        0/18         0/1    0/0     0/0      ❓ │   │       │       │   │       │   │   │   │   └── scopeguard 1.1.0
0/4        0/81         0/14   0/0     0/2      ❓ │   │       │       │   │       │   │   │   └── crossbeam-utils 0.8.11
0/0        0/0          0/0    0/0     0/0      ❓ │   │       │       │   │       │   │   ├── either 1.7.0
0/0        5/5          0/0    0/0     0/0      ☢  │   │       │       │   │       │   │   │   └── serde 1.0.140
0/5        0/488        0/2    0/0     0/20     ❓ │   │       │       │   │       │   │   └── rayon-core 1.9.3
0/2        0/494        0/7    0/0     0/14     ❓ │   │       │       │   │       │   │       ├── crossbeam-channel 0.5.6
0/0        0/0          0/0    0/0     0/0      ❓ │   │       │       │   │       │   │       │   ├── cfg-if 1.0.0
0/4        0/81         0/14   0/0     0/2      ❓ │   │       │       │   │       │   │       │   └── crossbeam-utils 0.8.11
0/0        0/453        0/6    0/0     0/6      ❓ │   │       │       │   │       │   │       ├── crossbeam-deque 0.8.2
0/4        0/81         0/14   0/0     0/2      ❓ │   │       │       │   │       │   │       ├── crossbeam-utils 0.8.11
0/4        0/81         0/14   0/0     0/2      ❓ │   │       │       │   │       │   │       ├── crossbeam-utils 0.8.11
0/0        0/72         0/0    0/0     0/0      ❓ │   │       │       │   │       │   │       └── num_cpus 1.13.1
1/21       10/368       0/2    0/0     5/40     ☢  │   │       │       │   │       │   │           └── libc 0.2.126
0/0        5/5          0/0    0/0     0/0      ☢  │   │       │       │   │       │   └── serde 1.0.140
0/6        0/663        0/5    0/0     0/3      ❓ │   │       │       │   │       ├── rayon 1.5.3
0/0        5/5          0/0    0/0     0/0      ☢  │   │       │       │   │       │   └── serde 1.0.140
0/0        0/0          0/18   0/2     0/0      ❓ │   │       │       │   │       └── stable_deref_trait 1.2.0
0/0        0/0          0/0    0/0     0/0      ❓ │   │       │       │   ├── rustc-demangle 0.1.21
0/1        0/399        0/7    0/1     0/13     ❓ │   │       │       │   └── smallvec 1.9.0
0/0        0/0          0/0    0/0     0/0      ❓ │   │       │       ├── cfg-if 1.0.0
1/21       10/368       0/2    0/0     5/40     ☢  │   │       │       ├── libc 0.2.126
0/0        0/0          0/0    0/0     0/0      🔒 │   │       │       ├── miniz_oxide 0.5.3
0/0        0/0          0/0    0/0     0/0      🔒 │   │       │       │   └── adler 1.0.2
0/0        0/26         0/0    0/1     0/0      ❓ │   │       │       ├── object 0.29.0
0/6        0/156        0/0    0/0     0/0      ❓ │   │       │       │   ├── crc32fast 1.3.2
0/0        0/0          0/0    0/0     0/0      ❓ │   │       │       │   │   └── cfg-if 1.0.0
0/2        0/118        0/2    0/0     0/2      ❓ │   │       │       │   ├── flate2 1.0.24
0/6        0/156        0/0    0/0     0/0      ❓ │   │       │       │   │   ├── crc32fast 1.3.2
0/0        0/0          0/0    0/0     0/0      🔒 │   │       │       │   │   └── miniz_oxide 0.5.3
0/1        0/1367       0/24   0/1     0/69     ❓ │   │       │       │   ├── hashbrown 0.12.3
0/0        0/46         0/1    0/0     0/0      ❓ │   │       │       │   ├── indexmap 1.9.1
36/37      2067/2144    0/0    0/0     21/21    ☢  │   │       │       │   └── memchr 2.5.0
1/21       10/368       0/2    0/0     5/40     ☢  │   │       │       │       └── libc 0.2.126
0/0        0/0          0/0    0/0     0/0      ❓ │   │       │       ├── rustc-demangle 0.1.21
0/0        5/5          0/0    0/0     0/0      ☢  │   │       │       └── serde 1.0.140
0/0        1/1          0/0    0/0     0/0      ☢  │   │       ├── bs58 0.3.1
0/0        0/0          0/0    0/0     0/0      ❓ │   │       ├── heck 0.3.3
0/0        0/0          0/0    0/0     0/0      ❓ │   │       │   └── unicode-segmentation 1.9.0
0/0        12/12        0/0    0/0     3/3      ☢  │   │       ├── proc-macro2 1.0.41
0/0        0/0          0/0    0/0     0/0      ❓ │   │       ├── proc-macro2-diagnostics 0.9.1
0/0        12/12        0/0    0/0     3/3      ☢  │   │       │   ├── proc-macro2 1.0.41
0/0        0/0          0/0    0/0     0/0      ❓ │   │       │   ├── quote 1.0.20
0/0        50/50        3/3    0/0     2/2      ☢  │   │       │   ├── syn 1.0.98
3/3        34/34        0/0    0/0     0/0      ☢  │   │       │   └── yansi 0.5.1
0/0        0/0          0/0    0/0     0/0      ❓ │   │       ├── quote 1.0.20
0/0        5/5          0/0    0/0     0/0      ☢  │   │       ├── serde 1.0.140
0/0        4/7          0/0    0/0     0/0      ☢  │   │       ├── serde_json 1.0.82
0/0        0/46         0/1    0/0     0/0      ❓ │   │       │   ├── indexmap 1.9.1
0/0        7/7          0/0    0/0     0/0      ☢  │   │       │   ├── itoa 1.0.2
7/9        587/723      0/0    0/0     2/2      ☢  │   │       │   ├── ryu 1.0.10
0/0        5/5          0/0    0/0     0/0      ☢  │   │       │   └── serde 1.0.140
8/8        202/202      0/0    0/0     0/0      ☢  │   │       ├── sha2 0.9.9
0/0        6/6          0/0    0/0     0/0      ☢  │   │       │   ├── block-buffer 0.9.0
0/0        3/3          0/0    0/0     0/0      ☢  │   │       │   │   ├── block-padding 0.2.1
1/1        292/292      20/20  8/8     5/5      ☢  │   │       │   │   └── generic-array 0.14.5
0/0        5/5          0/0    0/0     0/0      ☢  │   │       │   │       ├── serde 1.0.140
0/0        0/0          0/0    0/0     0/0      🔒 │   │       │   │       └── typenum 1.15.0
0/0        0/0          0/0    0/0     0/0      ❓ │   │       │   ├── cfg-if 1.0.0
0/1        0/14         0/0    0/0     0/0      ❓ │   │       │   ├── cpufeatures 0.2.2
0/0        0/0          0/0    0/0     0/0      🔒 │   │       │   ├── digest 0.9.0
1/1        292/292      20/20  8/8     5/5      ☢  │   │       │   │   └── generic-array 0.14.5
0/0        0/0          0/0    0/0     0/0      ❓ │   │       │   └── opaque-debug 0.3.0
```

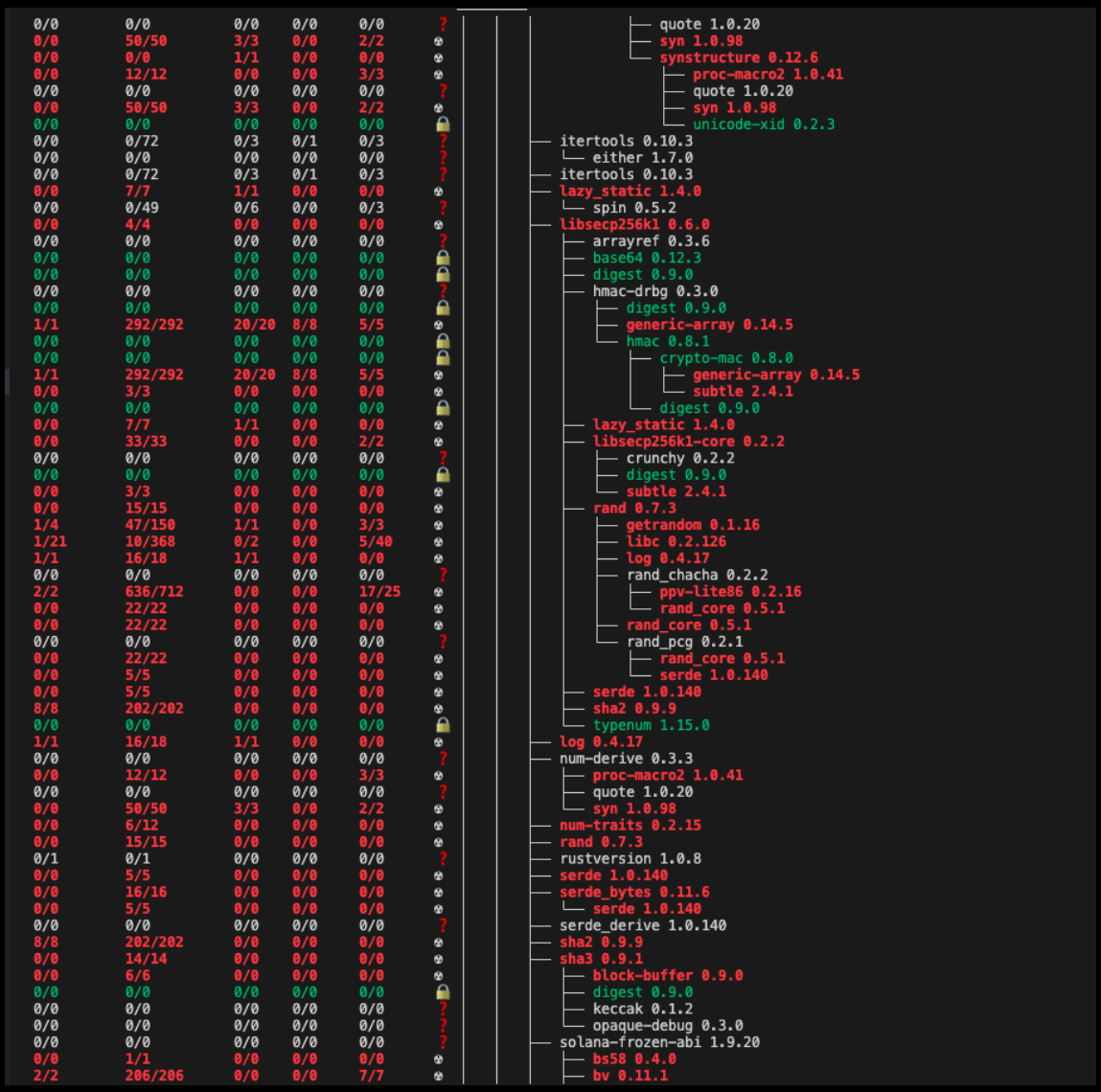
AUTOMATED TESTING

```
36/37   2067/2144  0/0  0/0  21/21  ☢  │   │   │   │   └── memchr 2.5.0
36/37   2067/2144  0/0  0/0  21/21  ☢  │   │   │   ├── memchr 2.5.0
0/0     0/0        0/0  0/0  0/0    🔒  │   │   │   └── regex-syntax 0.6.27
0/0     50/50      3/3  0/0  2/2    ☢  │   │   └── syn 1.0.98
0/0     0/0        0/0  0/0  0/0    ?  │   ├── anchor-attribute-account 0.24.2
0/0     8/8        0/0  0/0  0/0    ☢  │   │   ├── anchor-syn 0.24.2
15/18   442/449    3/3  0/0  11/11  ☢  │   │   ├── anyhow 1.0.58
0/0     1/1        0/0  0/0  0/0    ☢  │   │   ├── bs58 0.4.0
8/8     202/202    0/0  0/0  0/0    ☢  │   │   │   └── sha2 0.9.9
0/0     12/12      0/0  0/0  3/3    ☢  │   │   ├── proc-macro2 1.0.41
0/0     0/0        0/0  0/0  0/0    ?  │   │   ├── quote 1.0.20
0/1     0/1        0/0  0/0  0/0    ?  │   │   ├── rustversion 1.0.8
0/0     50/50      3/3  0/0  2/2    ☢  │   │   └── syn 1.0.98
0/0     0/0        0/0  0/0  0/0    ?  │   ├── anchor-attribute-constant 0.24.2
0/0     8/8        0/0  0/0  0/0    ☢  │   │   ├── anchor-syn 0.24.2
0/0     12/12      0/0  0/0  3/3    ☢  │   │   ├── proc-macro2 1.0.41
0/0     50/50      3/3  0/0  2/2    ☢  │   │   └── syn 1.0.98
0/0     0/0        0/0  0/0  0/0    ?  │   ├── anchor-attribute-error 0.24.2
0/0     8/8        0/0  0/0  0/0    ☢  │   │   ├── anchor-syn 0.24.2
0/0     12/12      0/0  0/0  3/3    ☢  │   │   ├── proc-macro2 1.0.41
0/0     0/0        0/0  0/0  0/0    ?  │   │   ├── quote 1.0.20
0/0     50/50      3/3  0/0  2/2    ☢  │   │   └── syn 1.0.98
0/0     0/0        0/0  0/0  0/0    ?  │   ├── anchor-attribute-event 0.24.2
0/0     8/8        0/0  0/0  0/0    ☢  │   │   ├── anchor-syn 0.24.2
15/18   442/449    3/3  0/0  11/11  ☢  │   │   ├── anyhow 1.0.58
0/0     12/12      0/0  0/0  3/3    ☢  │   │   ├── proc-macro2 1.0.41
0/0     0/0        0/0  0/0  0/0    ?  │   │   ├── quote 1.0.20
0/0     50/50      3/3  0/0  2/2    ☢  │   │   └── syn 1.0.98
0/0     0/0        0/0  0/0  0/0    ?  │   ├── anchor-attribute-interface 0.24.2
0/0     8/8        0/0  0/0  0/0    ☢  │   │   ├── anchor-syn 0.24.2
15/18   442/449    3/3  0/0  11/11  ☢  │   │   ├── anyhow 1.0.58
0/0     0/0        0/0  0/0  0/0    ?  │   │   ├── heck 0.3.3
0/0     12/12      0/0  0/0  3/3    ☢  │   │   ├── proc-macro2 1.0.41
0/0     0/0        0/0  0/0  0/0    ?  │   │   ├── quote 1.0.20
0/0     50/50      3/3  0/0  2/2    ☢  │   │   └── syn 1.0.98
0/0     0/0        0/0  0/0  0/0    ?  │   ├── anchor-attribute-program 0.24.2
0/0     8/8        0/0  0/0  0/0    ☢  │   │   ├── anchor-syn 0.24.2
15/18   442/449    3/3  0/0  11/11  ☢  │   │   ├── anyhow 1.0.58
0/0     12/12      0/0  0/0  3/3    ☢  │   │   ├── proc-macro2 1.0.41
0/0     0/0        0/0  0/0  0/0    ?  │   │   ├── quote 1.0.20
0/0     50/50      3/3  0/0  2/2    ☢  │   │   └── syn 1.0.98
0/0     0/0        0/0  0/0  0/0    ?  │   ├── anchor-attribute-state 0.24.2
0/0     8/8        0/0  0/0  0/0    ☢  │   │   ├── anchor-syn 0.24.2
15/18   442/449    3/3  0/0  11/11  ☢  │   │   ├── anyhow 1.0.58
0/0     12/12      0/0  0/0  3/3    ☢  │   │   ├── proc-macro2 1.0.41
0/0     0/0        0/0  0/0  0/0    ?  │   │   ├── quote 1.0.20
0/0     50/50      3/3  0/0  2/2    ☢  │   │   └── syn 1.0.98
0/0     0/0        0/0  0/0  0/0    ?  │   ├── anchor-derive-accounts 0.24.2
0/0     8/8        0/0  0/0  0/0    ☢  │   │   ├── anchor-syn 0.24.2
15/18   442/449    3/3  0/0  11/11  ☢  │   │   ├── anyhow 1.0.58
0/0     12/12      0/0  0/0  3/3    ☢  │   │   ├── proc-macro2 1.0.41
0/0     0/0        0/0  0/0  0/0    ?  │   │   ├── quote 1.0.20
0/0     50/50      3/3  0/0  2/2    ☢  │   │   └── syn 1.0.98
0/0     0/0        0/0  0/0  0/0    ?  │   ├── arrayref 0.3.6
0/0     0/0        0/0  0/0  0/0    🔒  │   ├── base64 0.13.0
0/0     22/22      0/0  0/0  0/0    ☢  │   ├── bincode 1.3.3
0/0     5/5        0/0  0/0  0/0    ☢  │   │   └── serde 1.0.140
0/0     7/7        0/0  0/0  0/0    ☢  │   ├── borsh 0.9.3
0/0     0/0        0/0  0/0  0/0    ?  │   │   ├── borsh-derive 0.9.3
0/0     0/0        0/0  0/0  0/0    ?  │   │   │   ├── borsh-derive-internal 0.9.3
0/0     12/12      0/0  0/0  3/3    ☢  │   │   │   │   ├── proc-macro2 1.0.41
0/0     0/0        0/0  0/0  0/0    ?  │   │   │   │   ├── quote 1.0.20
0/0     50/50      3/3  0/0  2/2    ☢  │   │   │   │   └── syn 1.0.98
0/0     0/0        0/0  0/0  0/0    ?  │   │   │   ├── borsh-schema-derive-internal 0.9.3
```

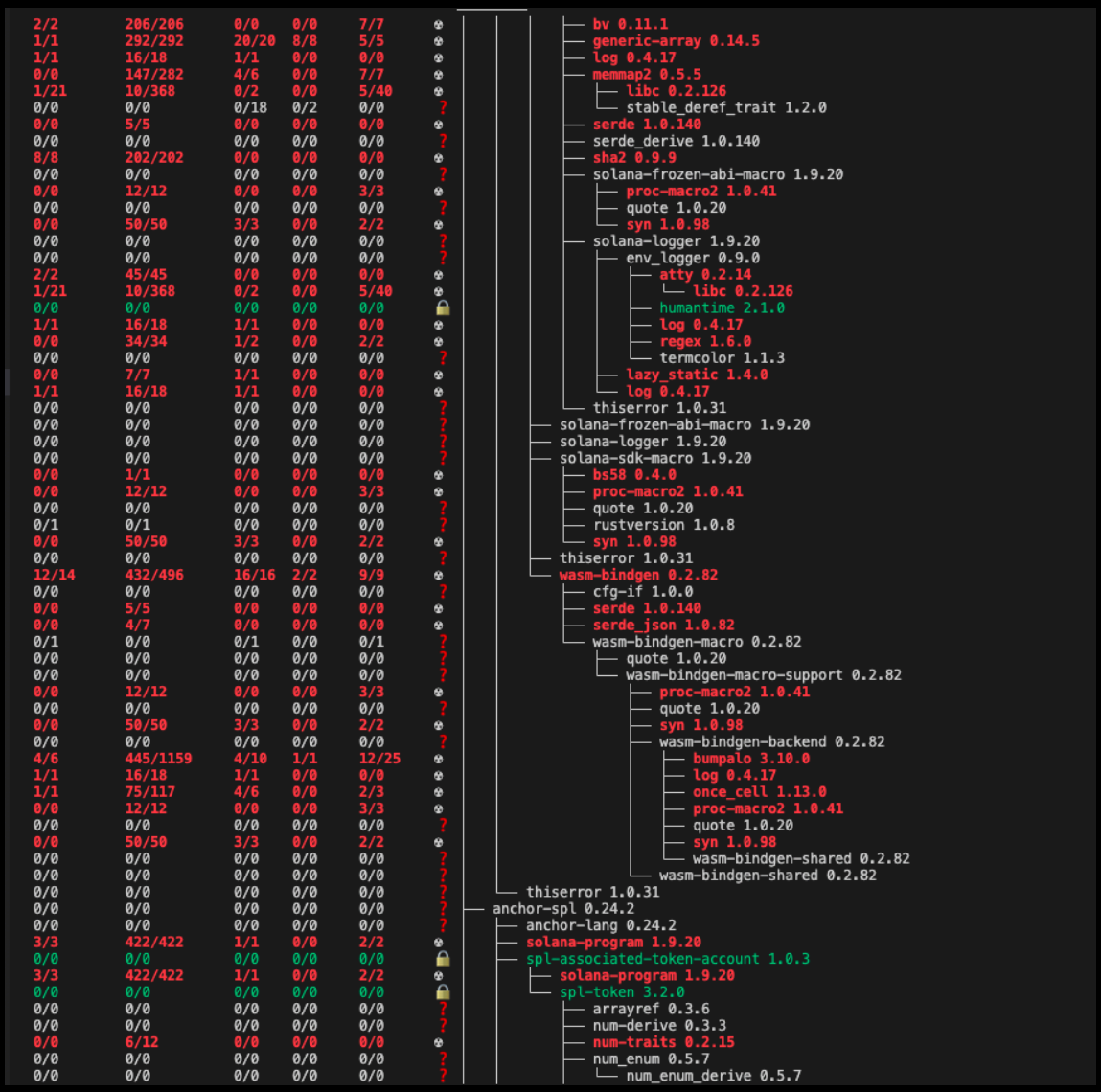
AUTOMATED TESTING



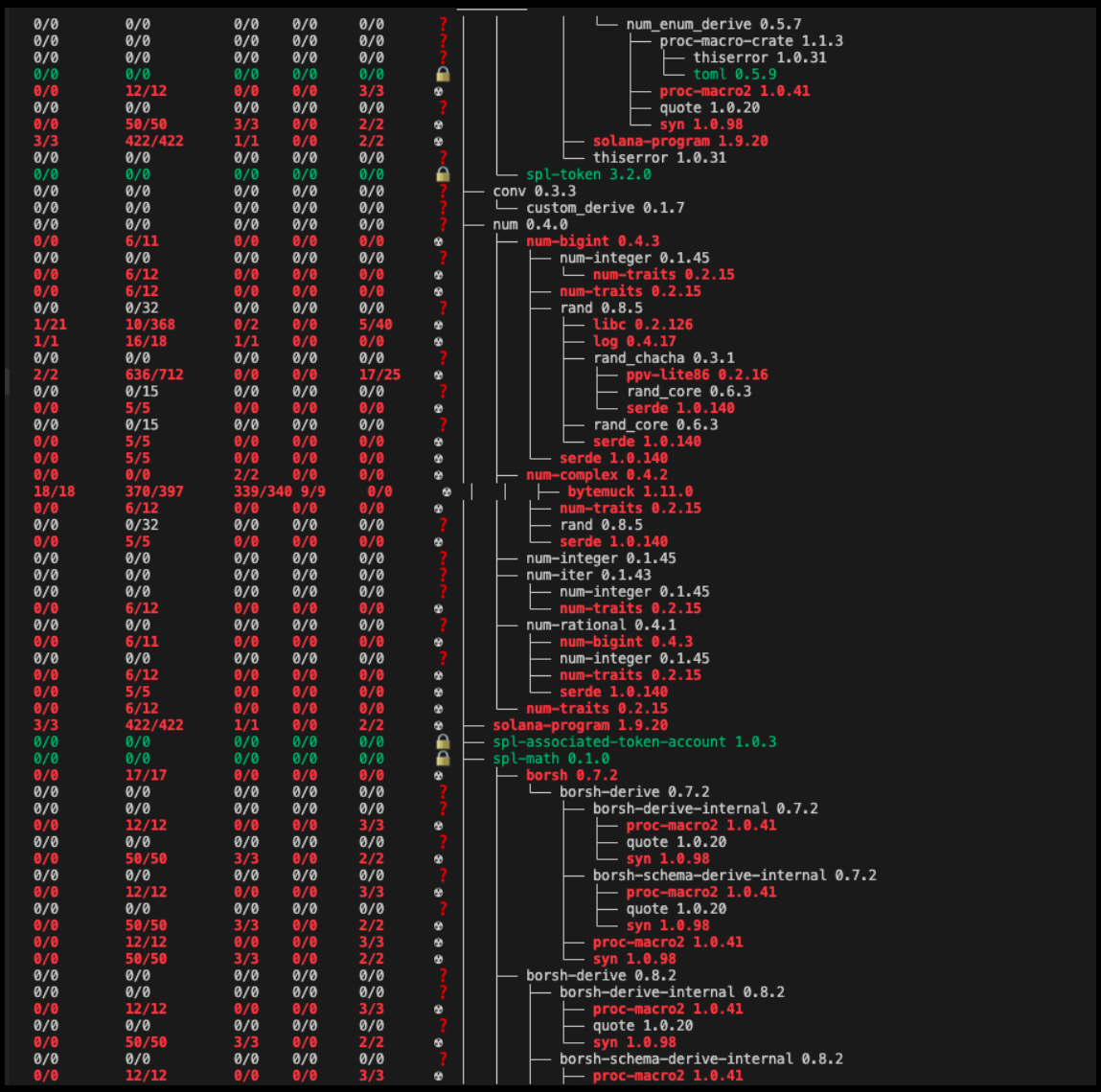
AUTOMATED TESTING



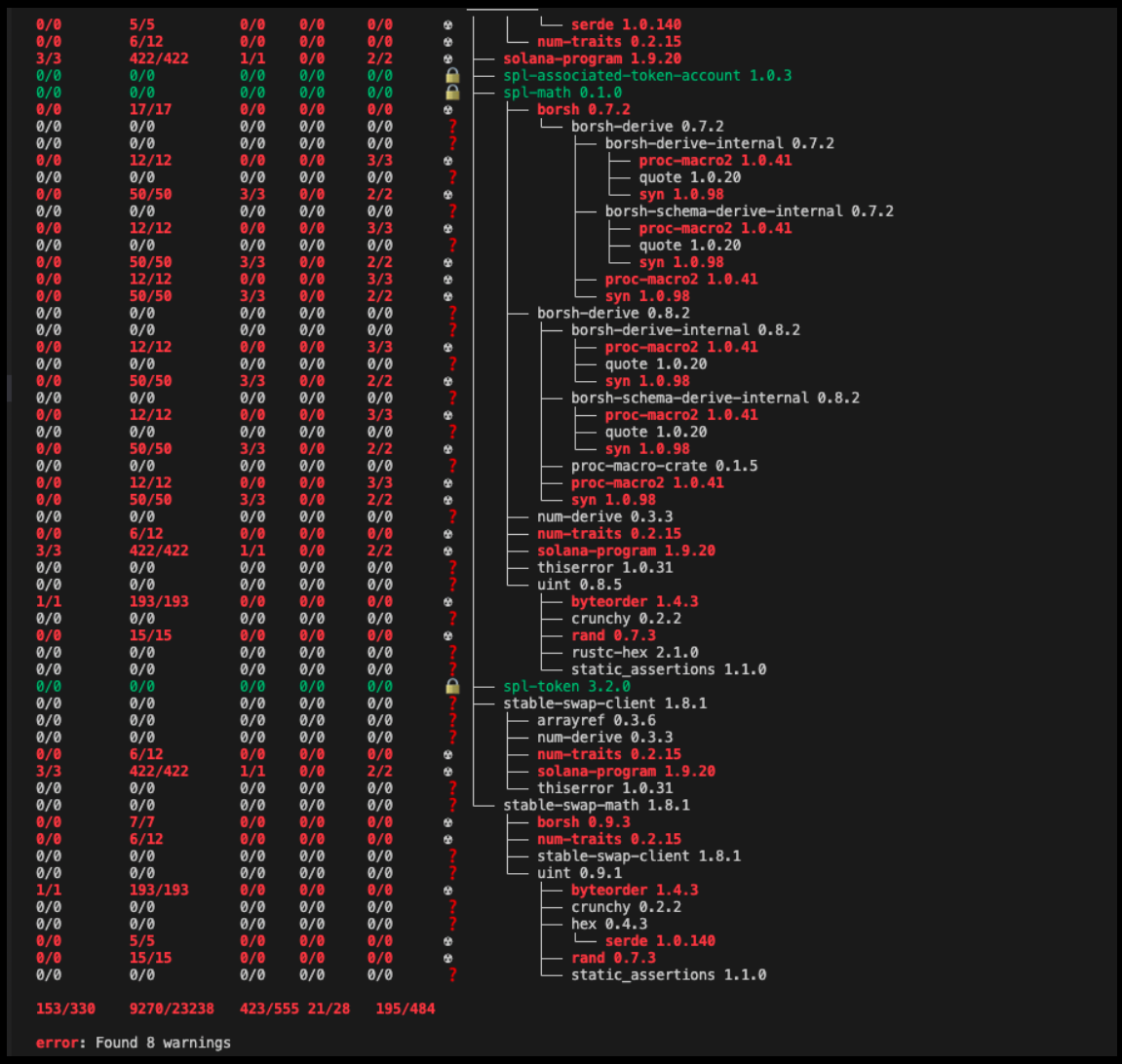
AUTOMATED TESTING



AUTOMATED TESTING



AUTOMATED TESTING



```
0/0         5/5         0/0    0/0    0/0      ☢  │   │   └── serde 1.0.140
0/0         6/12        0/0    0/0    0/0      ☢  │   └── num-traits 0.2.15
3/3         422/422     1/1    0/0    2/2      ☢  ├── solana-program 1.9.20
0/0         0/0         0/0    0/0    0/0      🔒 ├── spl-associated-token-account 1.0.3
0/0         0/0         0/0    0/0    0/0      🔒 ├── spl-math 0.1.0
0/0         17/17       0/0    0/0    0/0      ☢  │   ├── borsh 0.7.2
0/0         0/0         0/0    0/0    0/0      ?  │   │   └── borsh-derive 0.7.2
0/0         0/0         0/0    0/0    0/0      ?  │   │       ├── borsh-derive-internal 0.7.2
0/0         12/12       0/0    0/0    3/3      ☢  │   │       │   ├── proc-macro2 1.0.41
0/0         0/0         0/0    0/0    0/0      ?  │   │       │   ├── quote 1.0.20
0/0         50/50       3/3    0/0    2/2      ☢  │   │       │   └── syn 1.0.98
0/0         0/0         0/0    0/0    0/0      ?  │   │       ├── borsh-schema-derive-internal 0.7.2
0/0         12/12       0/0    0/0    3/3      ☢  │   │       │   ├── proc-macro2 1.0.41
0/0         0/0         0/0    0/0    0/0      ?  │   │       │   ├── quote 1.0.20
0/0         50/50       3/3    0/0    2/2      ☢  │   │       │   └── syn 1.0.98
0/0         12/12       0/0    0/0    3/3      ☢  │   │       ├── proc-macro2 1.0.41
0/0         50/50       3/3    0/0    2/2      ☢  │   │       └── syn 1.0.98
0/0         0/0         0/0    0/0    0/0      ?  │   ├── borsh-derive 0.8.2
0/0         0/0         0/0    0/0    0/0      ?  │   │   ├── borsh-derive-internal 0.8.2
0/0         12/12       0/0    0/0    3/3      ☢  │   │   │   ├── proc-macro2 1.0.41
0/0         0/0         0/0    0/0    0/0      ?  │   │   │   ├── quote 1.0.20
0/0         50/50       3/3    0/0    2/2      ☢  │   │   │   └── syn 1.0.98
0/0         0/0         0/0    0/0    0/0      ?  │   │   ├── borsh-schema-derive-internal 0.8.2
0/0         12/12       0/0    0/0    3/3      ☢  │   │   │   ├── proc-macro2 1.0.41
0/0         0/0         0/0    0/0    0/0      ?  │   │   │   ├── quote 1.0.20
0/0         50/50       3/3    0/0    2/2      ☢  │   │   │   └── syn 1.0.98
0/0         0/0         0/0    0/0    0/0      ?  │   │   ├── proc-macro-crate 0.1.5
0/0         12/12       0/0    0/0    3/3      ☢  │   │   ├── proc-macro2 1.0.41
0/0         50/50       3/3    0/0    2/2      ☢  │   │   └── syn 1.0.98
0/0         0/0         0/0    0/0    0/0      ?  │   ├── num-derive 0.3.3
0/0         6/12        0/0    0/0    0/0      ☢  │   ├── num-traits 0.2.15
3/3         422/422     1/1    0/0    2/2      ☢  │   ├── solana-program 1.9.20
0/0         0/0         0/0    0/0    0/0      ?  │   ├── thiserror 1.0.31
0/0         0/0         0/0    0/0    0/0      ?  │   └── uint 0.8.5
1/1         193/193     0/0    0/0    0/0      ☢  │       ├── byteorder 1.4.3
0/0         0/0         0/0    0/0    0/0      ?  │       ├── crunchy 0.2.2
0/0         15/15       0/0    0/0    0/0      ☢  │       ├── rand 0.7.3
0/0         0/0         0/0    0/0    0/0      ?  │       ├── rustc-hex 2.1.0
0/0         0/0         0/0    0/0    0/0      ?  │       └── static_assertions 1.1.0
0/0         0/0         0/0    0/0    0/0      🔒 ├── spl-token 3.2.0
0/0         0/0         0/0    0/0    0/0      ?  ├── stable-swap-client 1.8.1
0/0         0/0         0/0    0/0    0/0      ?  │   ├── arrayref 0.3.6
0/0         0/0         0/0    0/0    0/0      ?  │   ├── num-derive 0.3.3
0/0         6/12        0/0    0/0    0/0      ☢  │   ├── num-traits 0.2.15
3/3         422/422     1/1    0/0    2/2      ☢  │   ├── solana-program 1.9.20
0/0         0/0         0/0    0/0    0/0      ?  │   └── thiserror 1.0.31
0/0         0/0         0/0    0/0    0/0      ?  └── stable-swap-math 1.8.1
0/0         7/7         0/0    0/0    0/0      ☢      ├── borsh 0.9.3
0/0         6/12        0/0    0/0    0/0      ☢      ├── num-traits 0.2.15
0/0         0/0         0/0    0/0    0/0      ?      ├── stable-swap-client 1.8.1
0/0         0/0         0/0    0/0    0/0      ?      └── uint 0.9.1
1/1         193/193     0/0    0/0    0/0      ☢          ├── byteorder 1.4.3
0/0         0/0         0/0    0/0    0/0      ?          ├── crunchy 0.2.2
0/0         0/0         0/0    0/0    0/0      ?          ├── hex 0.4.3
0/0         5/5         0/0    0/0    0/0      ☢          │   └── serde 1.0.140
0/0         15/15       0/0    0/0    0/0      ☢          ├── rand 0.7.3
0/0         0/0         0/0    0/0    0/0      ?          └── static_assertions 1.1.0

153/330     9270/23238   423/555 21/28   195/484

error: Found 8 warnings
```

THANK YOU FOR CHOOSING

// HALBORN