Coral Cube - MMM

Solana Program Security Audit

Prepared by: **Halborn**

Date of Engagement: **October 21st, 2022 - November 18th, 2022**

Visit: **Halborn.com**

DOCUMENT REVISION HISTORY

VERSION	MODIFICATION	DATE	AUTHOR
0.1	Document Creation	11/18/2022	Pablo Gomez
0.2	Final Draft	11/18/2022	Pablo Gomez
0.3	Draft Review	11/18/2022	Piotr Cielas
0.4	Draft Review	11/18/2022	Gabi Urrutia
1.0	Remediation Plan	11/25/2022	Pablo Gomez
1.1	Remediation Plan Review	11/25/2022	Isabel Burruezo
1.2	Remediation Plan Review	11/25/2022	Piotr Cielas
1.3	Remediation Plan Review	11/25/2022	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
Piotr Cielas	Halborn	Piotr.Cielas@halborn.com
Isabel Burruezo	Halborn	Isabel.Burruezo@halborn.com
Pablo Gomez	Halborn	Pablo.Gomez@halborn.com

EXECUTIVE OVERVIEW

1.1 INTRODUCTION

Coral Cube engaged Halborn to conduct a security audit on their Solana programs, beginning on October 21st, 2022 and ending on November 18th, 2022 . The security assessment was scoped to the programs provided in the Coral Cube GitHub repository. Commit hashes and further details can be found in the Scope section of this report.

Coral Cube MMM is a multi-asset NFT and SFT AMM platform. Users can create pools and trade NFTs and SFTs in exchange for SOL.

1.2 AUDIT SUMMARY

The team at Halborn was provided three 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 Coral Cube . The main ones are the following:

- Set further checks at the time of pool creations in order to prevent irregular reverts and transaction failures

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.

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL

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

1.4 SCOPE

Code repositories:

1. Coral Cube MMM

- Repository: mmm
- Commit ID: 99c54c649adfe1c1668766fdb218e8841a6e949f
- Fixes Commit ID: 892c5041e616c56fc531f6cb047568532ced1db9
- Programs in scope:
    1. MMM (programs/mmm)

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

2. ASSESSMENT SUMMARY & FINDINGS OVERVIEW

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL
0	0	0	4	2

LIKELIHOOD

IMPACT

(HAL-02) (HAL-03)	(HAL-01)			
(HAL-06)				
(HAL-05)		(HAL-04)		

SECURITY ANALYSIS	RISK LEVEL	REMEDIATION DATE
(HAL-01) - UNUSABLE POOLS WITH ARBITRARY PAYMENT MINTS	Low	SOLVED - 11/18/2022
(HAL-02) - LIMITED-DEPOSIT POOLS	Low	SOLVED - 11/18/2022
(HAL-03) - FEES NOT CONTROLLED	Low	SOLVED - 11/18/2022
(HAL-04) - SPOT PRICE CAN BE SET TO ZERO	Low	SOLVED - 11/12/2022
(HAL-05) - COMPILATION ERRORS DUE TO INCORRECT DEPENDENCY VERSION	Informational	SOLVED - 11/18/2022
(HAL-06) - MISCONFIGURED SPOT PRICE AND DELTA CAN LEAD TO AN UNUSABLE POOL	Informational	ACKNOWLEDGED

FINDINGS & TECH DETAILS

3.1 (HAL-01) UNUSABLE POOLS WITH ARBITRARY PAYMENT MINTS - LOW

Description:

The pool owner may specify the payment mint address at the time of pool creation to support payments in arbitrary SPL tokens.

When the pool owner sets the payment mint address to an arbitrary address (besides the default address `Pubkey::default()`) the pool is unusable and users cannot trade NFTs/SFTs as all operations are reverted with the `InvalidPaymentMint` error. The pool cannot be closed or updated, and so the rent is permanently locked in the pool account.

Code Location:

Listing 1: programs/mmm/src/instructions/create_pool.rs (Line 82)

```
71 ...
72
73     // state variables
74     pool.sellside_asset_amount = 0;
75     pool.buyside_payment_amount = 0;
76     pool.lp_fee_earned = 0;
77
78     // immutable
79     pool.owner = owner.key();
80     pool.cosigner = cosigner.key();
81     pool.uuid = args.uuid;
82     pool.payment_mint = args.payment_mint;
83     pool.allowlists = args.allowlists;
84
85     Ok(())
86 }
```

Listing 2: programs/mmm/src/instructions/sol_fulfill_sell.rs (Line 48)

```
42 #[account(
43     mut,
```

```
44      seeds = [POOL_PREFIX.as_bytes(), owner.key().as_ref(), pool.
↳ uuid.as_ref()],
45      has_one = owner @ MMMErrorCode::InvalidOwner,
46      has_one = referral @ MMMErrorCode::InvalidReferral,
47      has_one = cosigner @ MMMErrorCode::InvalidCosigner,
48      constraint = pool.payment_mint.eq(&Pubkey::default()) @
↳ MMMErrorCode::InvalidPaymentMint,
49      constraint = pool.expiry == 0 || pool.expiry > Clock::get().
↳ unwrap().unix_timestamp @ MMMErrorCode::Expired,
50      constraint = args.buyside_creator_royalty_bp <= 10000 @
↳ MMMErrorCode::InvalidBP,
51      bump
52  )]
53  pub pool: Box<Account<'info, Pool>>,
```

Risk Level:

Likelihood - 2
Impact - 3

Recommendation:

In the CreatePool instruction handler, ensure the payment mint address is equal to Pubkey::default(), or remove this argument from instruction data altogether.

References:

- Solana Token Specification

Remediation Plan:

SOLVED: The Coral Cube team implemented a constraint that checks if the given mint is the Default Pubkey at pool creation time as follows:

Listing 3

```
constraint = pool.payment_mint.eq(&Pubkey::default()) @
↳ MMMErrorCode::InvalidPaymentMint
```

Changes can be found in commit 892c5041e616c56fc531f6cb047568532ced1db9.

3.2 (HAL-02) LIMITED-DEPOSIT POOLS - LOW

Pool owner may specify a referral account which receives some share of pool handling fees. A pool owner can set their address as referral, so they can receive pool fees in full, that is owner fees and referral fees.

Listing 4: programs/mmm/src/instructions/sol_fulfill_buy.rs (Line 261)

```
260 if referral_fee > 0 {
261         anchor_lang::solana_program::program::invoke_signed(
262             &anchor_lang::solana_program::system_instruction::
 ↳ transfer(
263                 buyside_sol_escrow_account.key,
264                 referral.key,
265                 referral_fee,
266             ),
267             &[
268                 buyside_sol_escrow_account.to_account_info(),
269                 referral.to_account_info(),
270                 system_program.to_account_info(),
271             ],
272             buyside_sol_escrow_account_seeds,
273         )?;
274     }
```

This instruction throws the UnbalacedInstruction error when invoked by someone else than the owner, preventing third parties from selling their NFTs/SFTs to the pool.

Code Location:

- programs/mmm/instructions/sol_fulfill_buy.rs files

Risk Level:

Likelihood - 1
Impact - 3

Recommendation:

Throw more explicit error or implement a restriction on the referral property to prevent the owner to set their address as the referral

References:

- Unbalanced Instruction reference

Remediation Plan:

SOLVED: The Coral Cube team implemented a constraint that checks if the owner is the same as the referred one at the time of pool creation. Changes can be found in commit 892c5041e616c56fc531f6cb047568532ced1db9.

3.3 (HAL-03) UNUSABLE POOLS WITH MISCONFIGURED POOL FEES - LOW

LP Fees and Referral fees are transferred to creators and managers on every trade in the pool. The lp_fee and referral_bp parameters can be set from 0 to 10000. If any of them is set to the maximum allowed value (10000), the fees the payer is charged are equal to the value of the asset.

The fee calculation formula is given in the code snippet below.

Listing 5: programs/mmm/src/util.rs (Line 133)

```
132 pub fn get_sol_referral_fee(pool: &Pool, total_sol_price: u64) ->
 ↳ Result<u64> {
133     Ok((total_sol_price as u128)
134         .checked_mul(pool.referral_bp as u128)
135         .ok_or(MMMErrorCode::NumericOverflow)?
136         .checked_div(10000)
137         .ok_or(MMMErrorCode::NumericOverflow)? as u64)
138 }
```

This function's intention is to calculate the total referral fee, but when referral fee is set to 10000, the total returned fee is accurately the total_sol_price, making the total payment for the asset as 0, as it can be seen in the payment_amount calculation formula below

Listing 6: programs/mmm/src/instructions/sol_fulfill_buy.rs (Line 218)

```
218 let payment_amount = total_price
219         .checked_sub(lp_fee)
220         .ok_or(MMMErrorCode::NumericOverflow)?
221         .checked_sub(referral_fee)
222         .ok_or(MMMErrorCode::NumericOverflow)?;
223     if payment_amount < args.min_payment_amount {
224         return Err(MMMErrorCode::InvalidRequestedPrice.into());
225     }
```

In this case, if LP Fee is 0, payment_amount is set as 0, but if LP

Fee is greater than 0, this expression throws the NumericOverflow error, reverting the transaction.

This prevents users from selling their assets.

Risk Level:

Likelihood - 1
Impact - 3

Recommendation:

Ensure the sum of the pool's LP Fee BP and the Referral Fee BP are lesser than 10000.

Remediation Plan:

SOLVED: The Coral Cube team implemented constraints at pool creation time for LP Fee BB and have split the referral fee into two new arguments: taker and maker, at the moment to make a buy or sell. Changes are made in commit 892c5041e616c56fc531f6cb047568532ced1db9.

3.4 (HAL-04) SPOT PRICE CAN BE SET TO ZERO - LOW

Description:

The spot price is set by the owner at the moment of creating the pool. All subsequent prices and fees in the pool depend on this parameter, since it is the reference value for the pool

When the pool creator sets the spot price to zero, the pool is going to revert every sell or buy operation due to the `NumericOverflow` error thrown in the `get_sol_total_price_and_next_price` function.

Code Location:

As it can be seen, `spot_price` argument is directly assigned to `pool.spot_price` without any checks.

Listing 7: programs/mmm/src/instructions/create_pool.rs (Line 61)

```
52 pub fn handler(ctx: Context<CreatePool>, args: CreatePoolArgs) ->
↳ Result<()> {
53     let pool = &mut ctx.accounts.pool;
54     let owner = &ctx.accounts.owner;
55     let cosigner = &ctx.accounts.cosigner;
56
57     check_allowlists(&args.allowlists)?;
58     check_curve(args.curve_type, args.curve_delta)?;
59
60     // mutable
61     pool.spot_price = args.spot_price;
62     pool.curve_type = args.curve_type;
63     pool.curve_delta = args.curve_delta;
64     pool.reinvest_fulfill_buy = args.reinvest_fulfill_buy;
65     pool.reinvest_fulfill_sell = args.reinvest_fulfill_sell;
66     pool.expiry = args.expiry;
```

Risk Level:

Likelihood - 3
Impact - 1

Recommendation:

Verify the pool spot price is greater than 0 when creating or updating the pool.

References:

- Anchor Lib.rs Reference

Remediation Plan:

SOLVED: The Coral Cube team modified the account creation login in order to avoid setting spot_price to zero at pool creation time:

Listing 8

```
1 constraint = args.spot_price > 0 @ MMMErrorCode::InvalidSpotPrice,
```

Changes were implemented in commit 9b1e5cbabc66bf743db2a311ad74d95ca864d1fb.

3.5 (HAL-05) COMPILATION ERRORS DUE TO INCORRECT DEPENDENCY VERSION - INFORMATIONAL

Some errors may raise at the time of compilation, and they must be addressed before performing any release in order to ease deployment and forking of the projects. In this case, because of the spl-associated-token-account version the program is using (1.0.3), an error may be thrown at compile time. There's no instruction module in this version crate (reference here), but the program is using it in the following way:

Code Location:

Listing 9: programs/mmm/src/ata.rs (Line 7)

```
1 use crate::errors::MMMErrorCode;
2 use anchor_lang::{
3     prelude::*,
4     solana_program::program_pack::{IsInitialized, Pack},
5 };
6 use anchor_spl::associated_token::get_associated_token_address;
7 use spl_associated_token_account::instruction;
8
9 fn assert_owned_by(account: &AccountInfo, owner: &Pubkey) ->
↳ Result<()> {
10 ...
```

The crate is defined in the Cargo.toml file as follows:

Listing 10: programs/mmm/Cargo.toml (Line 24)

```
21 ...
22 anchor-spl = "0.25.0"
23 mpl-token-metadata = { version = "1.4.1", features = ["no-
↳ entrypoint"] }
24 spl-token = { version = "3.3.1",  features = ["no-entrypoint"] }
25 spl-associated-token-account = {version = "1.0.3", features = ["no
↳ -entrypoint"]}
```

Recommendation:

It is recommended to use the version implementing the instruction module (1.0.5) or change the use statement to import the instruction from v1.0.3 like so:

Listing 11

```
1 use spl_associated_token_account::create_associated_token_account;
```

Reference:

- Associated Token Account - v1.0.3
- Associated Token Account - v1.0.5

Remediation Plan:

SOLVED: The Coral Cube team modified the Cargo.toml file and the dependencies there in order to match the module used in the source code. This fix has been made in commit 892c5041e616c56fc531f6cb047568532ced1db9.

3.6 (HAL-06) MISCONFIGURED SPOT PRICE AND DELTA CAN LEAD TO AN UNUSABLE POOL - INFORMATIONAL

Description:

Spot Price and Delta values can be set to arbitrary values at the time of creating a Pool (with a 10000 limit in Delta when using exponential curve). It is possible to create a pool in which **fulfilling a buy** cannot be executed. Because of the formula `n*(2*p-(n-1)*delta)/2` , `2*p` must be always greater than `(n-1)*delta` and `p-n*delta` must be positive. Otherwise, the expression throws the `NumericOverflow`.

Code Location:

Listing 12: programs/mmm/src/util.rs (Line 163)

```
163     CURVE_KIND_LINEAR => {
164         // n*(2*p-(n-1)*delta)/2
165         let total_price = n
166             .checked_mul(
167                 p.checked_mul(2)
168                     .ok_or(MMMErrorCode::NumericOverflow)?
169                     .checked_sub(
170                         n.checked_sub(1)
171                             .ok_or(MMMErrorCode::NumericOverflow)?
172                             .checked_mul(delta)
173                             .ok_or(MMMErrorCode::NumericOverflow)
 ↳ ?,
174                     )
175                     .ok_or(MMMErrorCode::NumericOverflow)?,
176             )
177             .ok_or(MMMErrorCode::NumericOverflow)?
178             .checked_div(2)
179             .ok_or(MMMErrorCode::NumericOverflow)?;
180
181         // p - n * delta
182         let final_price = p
```

```
183             .checked_sub(n.checked_mul(delta).ok_or(MMMErrorCode::
 ↳ NumericOverflow)?)
184             .ok_or(MMMErrorCode::NumericOverflow)?;
185         Ok((total_price, final_price))
186     }
187     CURVE_KIND_EXP => {
188         // for loop to prevent overflow
189         let mut total_price: u64 = 0;
190         let mut curr_price: u128 = p as u128;
191         for _ in 0..n {
192             total_price = total_price
193                 .checked_add(curr_price as u64)
194                 .ok_or(MMMErrorCode::NumericOverflow)?;
195             curr_price = curr_price
196                 .checked_mul(10000)
197                 .ok_or(MMMErrorCode::NumericOverflow)?
198                 .checked_div(
199                     (delta as u128)
200                         .checked_add(10000)
201                         .ok_or(MMMErrorCode::NumericOverflow)?,
202                 )
203                 .ok_or(MMMErrorCode::NumericOverflow)?;
204         }
205         Ok((total_price, curr_price as u64))
206     }
207     _ => Err(MMMErrorCode::InvalidCurveType.into()),
208 }
```

A user can set a high delta (when creating the pool) and cause the program to throw errors every time someone tries to use it (because `p-n*delta < 0`). Since this check is not controlled when creating the pool, pools initialized with those parameter values are not functional until updated.

Code Location:

- programs/mmm/utils.rs

Risk Level:

Likelihood - 1

Impact - 2

Recommendation:

It is recommended to verify that `p >= delta` when creating/updating the pool. This check should be implemented in the `CreatePool` and `UpdatePool` instruction handlers.

Remediation Plan:

ACKNOWLEDGED: This is an edge case where the pool does not allow users to sell their NFTs (`fulfill_buy`), but does allow them to buy NFTs deposited by the pool owner and increase the final_price, so that other users can finally sell their NFTs. It cannot only be used if the pool owner does not deposit NFTs/SFTs. In addition, the pool can be updated with the `UpdatePool` instruction. The severity of this finding was downgraded to informational.

AUTOMATED TESTING

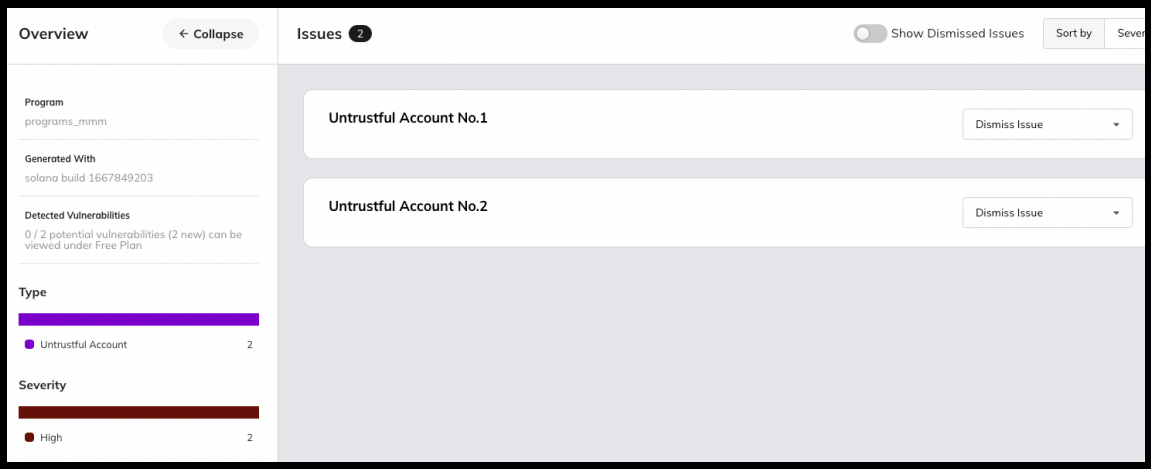
4.1 AUTOMATED VULNERABILITY SCANNING

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.

In this particular case, Soteria detected two potential issues related with Untrustful Accounts. These Accounts are the `allowlist_aux_account` ones. They are not being used at the time of reviewing the code, so the Soteria findings can be considered as false positives.

Results:



4.2 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:

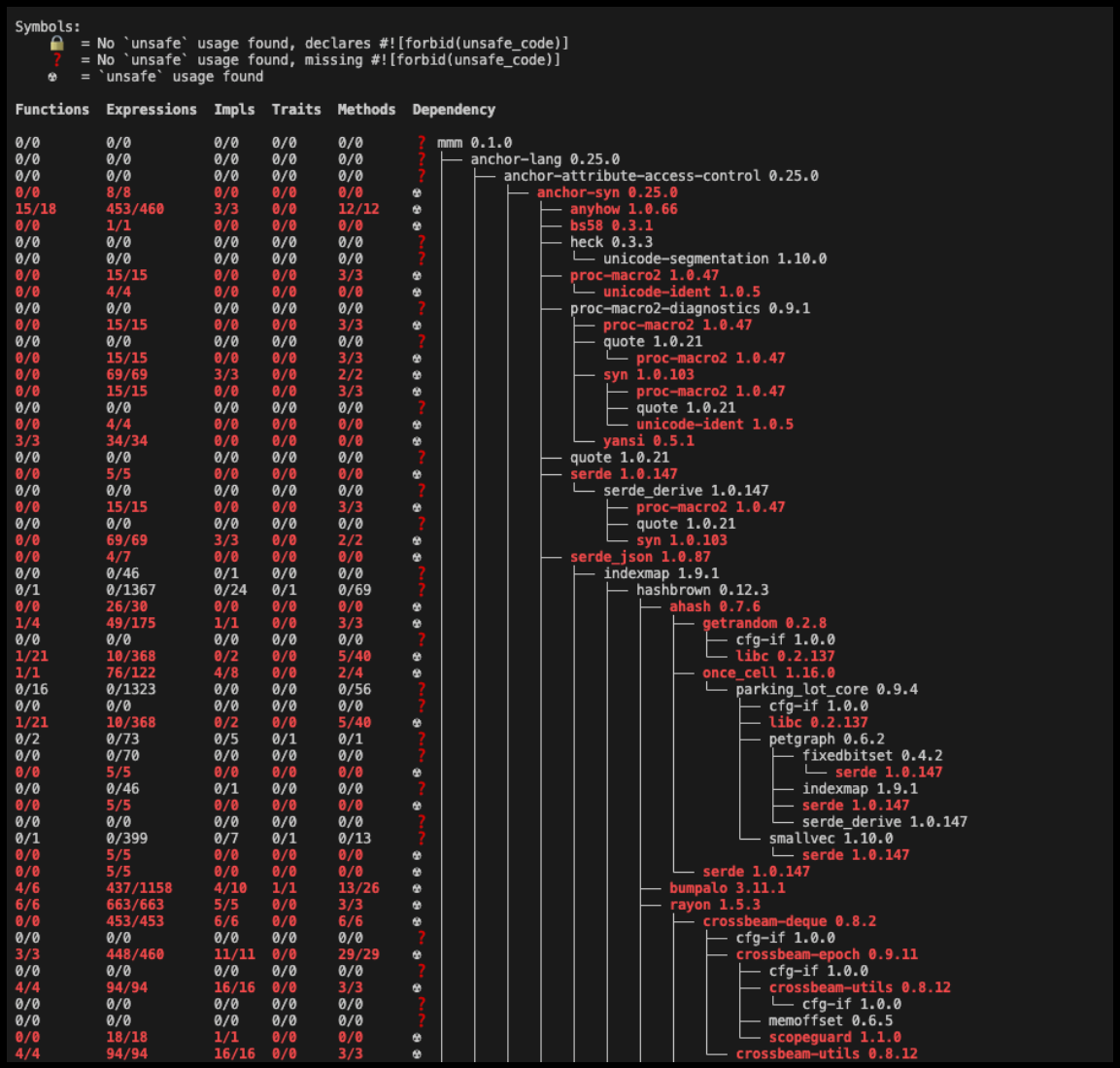
ID	package	Short Description
RUSTSEC-2020-0071	time	Potential segfault in the time crate

4.3 UNSAFE RUST CODE DETECTION

Description:

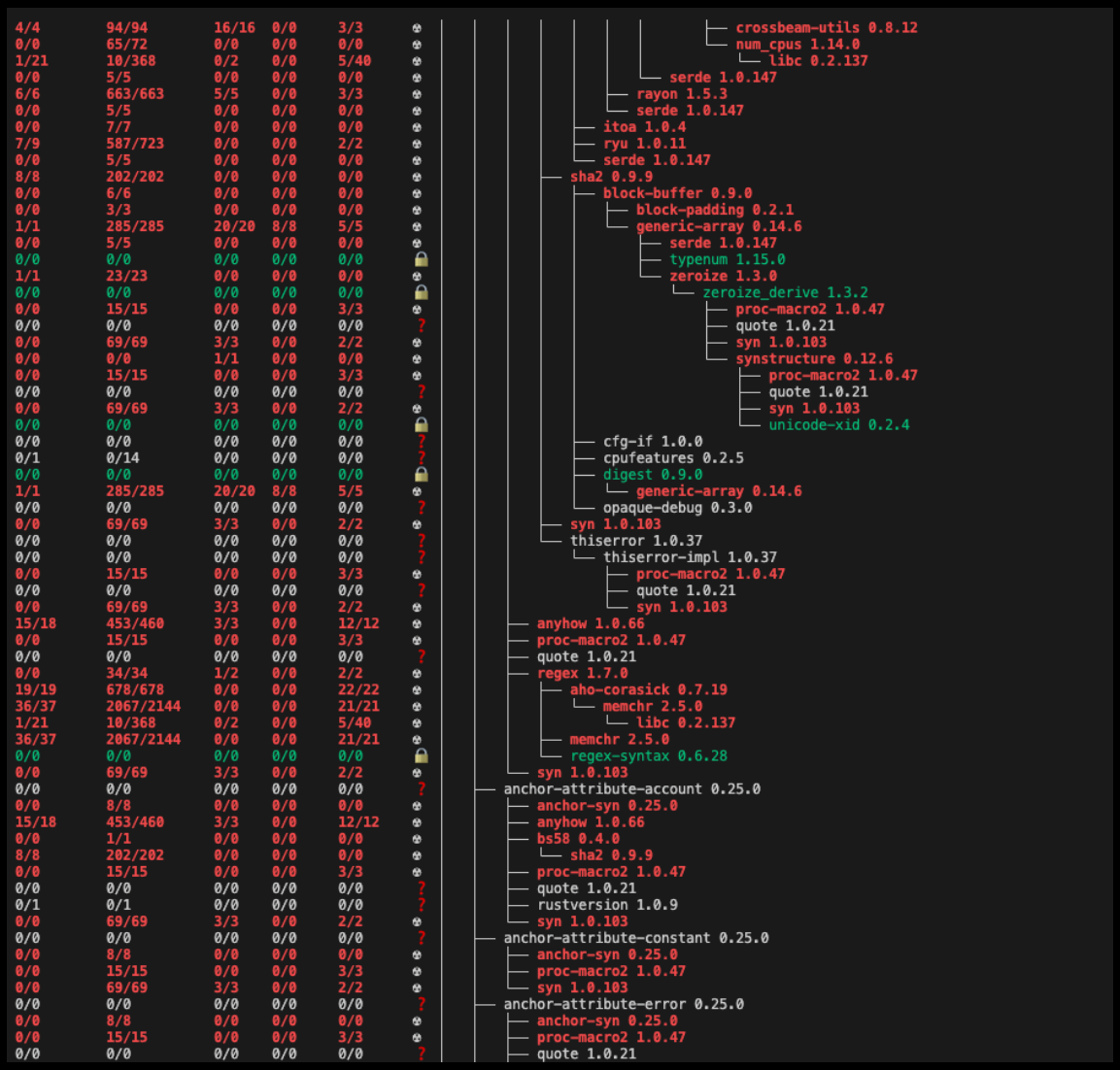
Halborn used automated security scanners to assist with the detection of well-known security issues and vulnerabilities. Among the tools used was cargo-geiger, a security tool that lists statistics related to the usage of unsafe Rust code in a core Rust codebase and all its dependencies.

MMM Results:

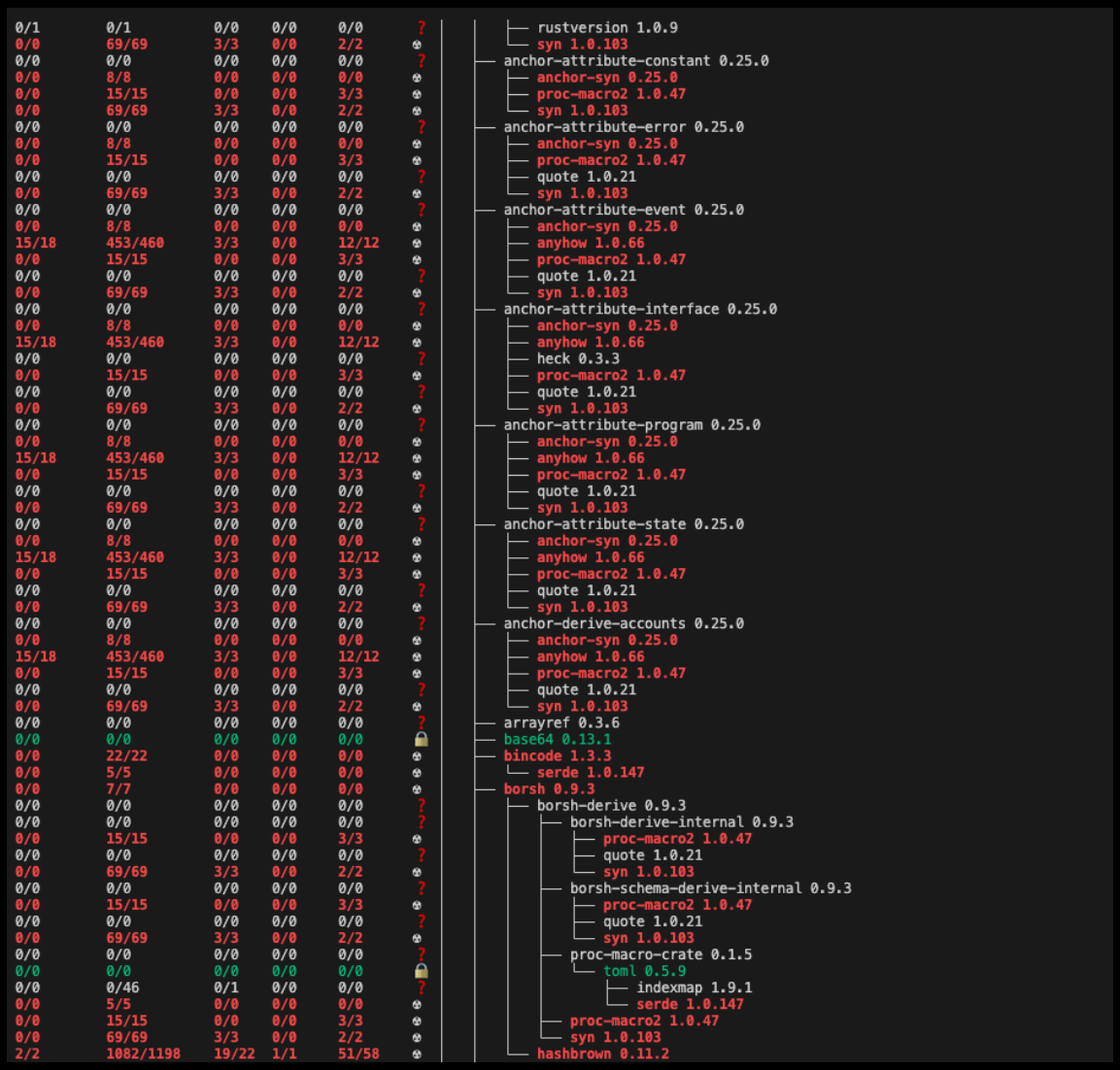


```
Symbols:
    🔒 = No `unsafe` usage found, declares #![forbid(unsafe_code)]
    ❓ = No `unsafe` usage found, missing #![forbid(unsafe_code)]
    ☢  = `unsafe` usage found

Functions  Expressions  Impls  Traits  Methods  Dependency

0/0        0/0          0/0    0/0     0/0      ❓ mmm 0.1.0
0/0        0/0          0/0    0/0     0/0      ❓ ├── anchor-lang 0.25.0
0/0        0/0          0/0    0/0     0/0      ❓ │   ├── anchor-attribute-access-control 0.25.0
0/0        8/8          0/0    0/0     0/0      ☢  │   │   ├── anchor-syn 0.25.0
15/18      453/460      3/3    0/0     12/12    ☢  │   │   │   ├── anyhow 1.0.66
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 1.10.0
0/0        15/15        0/0    0/0     3/3      ☢  │   │   │   ├── proc-macro2 1.0.47
0/0        4/4          0/0    0/0     0/0      ☢  │   │   │   │   └── unicode-ident 1.0.5
0/0        0/0          0/0    0/0     0/0      ❓ │   │   │   ├── proc-macro2-diagnostics 0.9.1
0/0        15/15        0/0    0/0     3/3      ☢  │   │   │   │   ├── proc-macro2 1.0.47
0/0        0/0          0/0    0/0     0/0      ❓ │   │   │   │   ├── quote 1.0.21
0/0        15/15        0/0    0/0     3/3      ☢  │   │   │   │   │   └── proc-macro2 1.0.47
0/0        69/69        3/3    0/0     2/2      ☢  │   │   │   │   ├── syn 1.0.103
0/0        15/15        0/0    0/0     3/3      ☢  │   │   │   │   │   ├── proc-macro2 1.0.47
0/0        0/0          0/0    0/0     0/0      ❓ │   │   │   │   │   ├── quote 1.0.21
0/0        4/4          0/0    0/0     0/0      ☢  │   │   │   │   │   └── unicode-ident 1.0.5
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 1.0.21
0/0        5/5          0/0    0/0     0/0      ☢  │   │   │   ├── serde 1.0.147
0/0        0/0          0/0    0/0     0/0      ❓ │   │   │   │   └── serde_derive 1.0.147
0/0        15/15        0/0    0/0     3/3      ☢  │   │   │   │       ├── proc-macro2 1.0.47
0/0        0/0          0/0    0/0     0/0      ❓ │   │   │   │       ├── quote 1.0.21
0/0        69/69        3/3    0/0     2/2      ☢  │   │   │   │       └── syn 1.0.103
0/0        4/7          0/0    0/0     0/0      ☢  │   │   │   ├── serde_json 1.0.87
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        49/175       1/1    0/0     3/3      ☢  │   │   │   │   │   │   │   ├── getrandom 0.2.8
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 0.2.137
1/1        76/122       4/8    0/0     2/4      ☢  │   │   │   │   │   │   │   ├── once_cell 1.16.0
0/16       0/1323       0/0    0/0     0/56     ❓ │   │   │   │   │   │   │   │   └── parking_lot_core 0.9.4
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 0.2.137
0/2        0/73         0/5    0/1     0/1      ❓ │   │   │   │   │   │   │   │       ├── petgraph 0.6.2
0/0        0/70         0/0    0/0     0/0      ❓ │   │   │   │   │   │   │   │       │   ├── fixedbitset 0.4.2
0/0        5/5          0/0    0/0     0/0      ☢  │   │   │   │   │   │   │   │       │   │   └── serde 1.0.147
0/0        0/46         0/1    0/0     0/0      ❓ │   │   │   │   │   │   │   │       │   ├── indexmap 1.9.1
0/0        5/5          0/0    0/0     0/0      ☢  │   │   │   │   │   │   │   │       │   ├── serde 1.0.147
0/0        0/0          0/0    0/0     0/0      ❓ │   │   │   │   │   │   │   │       │   └── serde_derive 1.0.147
0/1        0/399        0/7    0/1     0/13     ❓ │   │   │   │   │   │   │   │       └── smallvec 1.10.0
0/0        5/5          0/0    0/0     0/0      ☢  │   │   │   │   │   │   │   │           └── serde 1.0.147
0/0        5/5          0/0    0/0     0/0      ☢  │   │   │   │   │   │   │   └── serde 1.0.147
4/6        437/1158     4/10   1/1     13/26    ☢  │   │   │   │   │   │   ├── bumpalo 3.11.1
6/6        663/663      5/5    0/0     3/3      ☢  │   │   │   │   │   │   ├── rayon 1.5.3
0/0        453/453      6/6    0/0     6/6      ☢  │   │   │   │   │   │   │   ├── crossbeam-deque 0.8.2
0/0        0/0          0/0    0/0     0/0      ❓ │   │   │   │   │   │   │   │   ├── cfg-if 1.0.0
3/3        448/460      11/11  0/0     29/29    ☢  │   │   │   │   │   │   │   │   ├── crossbeam-epoch 0.9.11
0/0        0/0          0/0    0/0     0/0      ❓ │   │   │   │   │   │   │   │   │   ├── cfg-if 1.0.0
4/4        94/94        16/16  0/0     3/3      ☢  │   │   │   │   │   │   │   │   │   ├── crossbeam-utils 0.8.12
0/0        0/0          0/0    0/0     0/0      ❓ │   │   │   │   │   │   │   │   │   │   └── cfg-if 1.0.0
0/0        0/0          0/0    0/0     0/0      ❓ │   │   │   │   │   │   │   │   │   ├── memoffset 0.6.5
0/0        18/18        1/1    0/0     0/0      ☢  │   │   │   │   │   │   │   │   │   └── scopeguard 1.1.0
4/4        94/94        16/16  0/0     3/3      ☢  │   │   │   │   │   │   │   │   └── crossbeam-utils 0.8.12
```

AUTOMATED TESTING

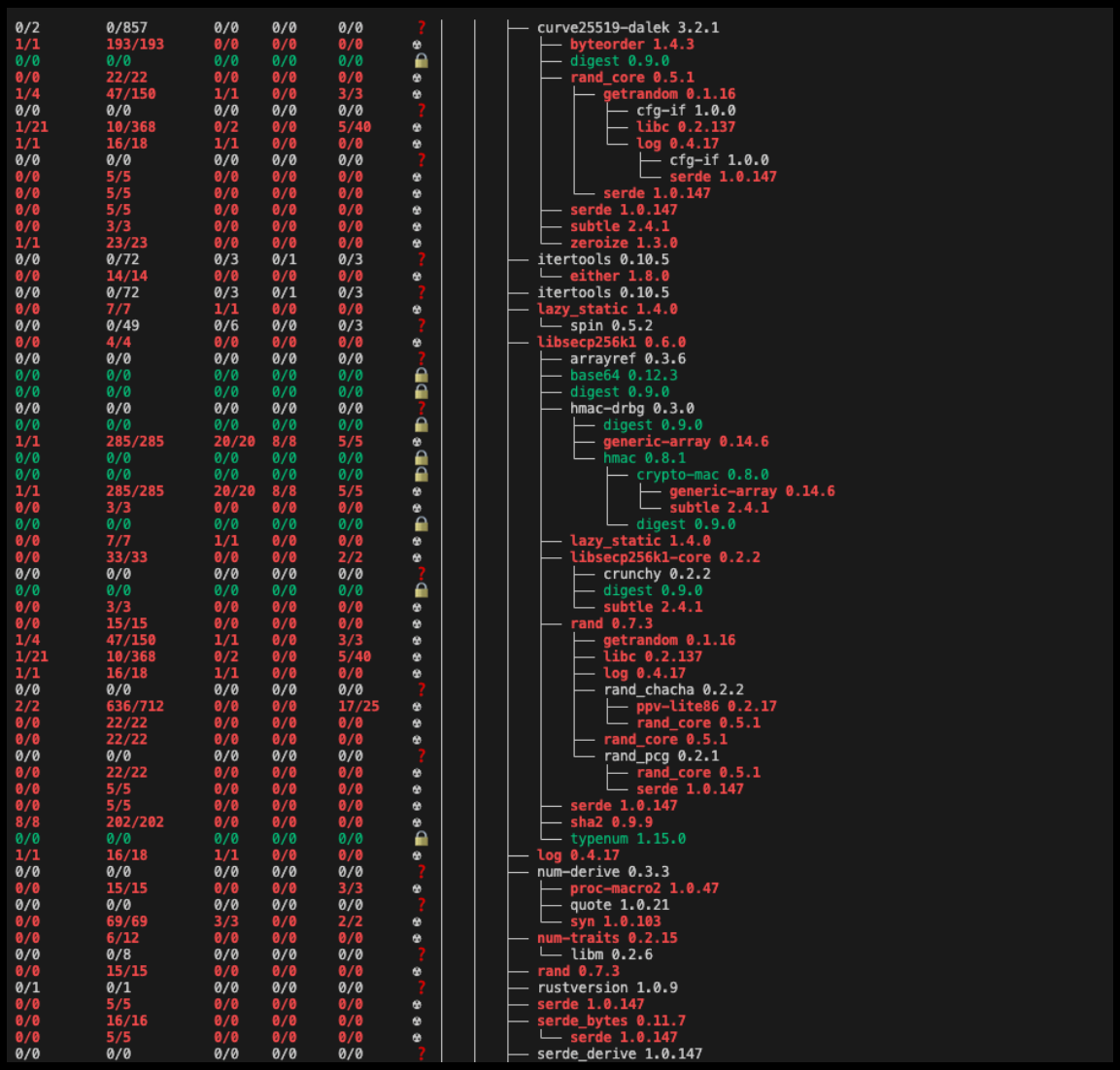


AUTOMATED TESTING

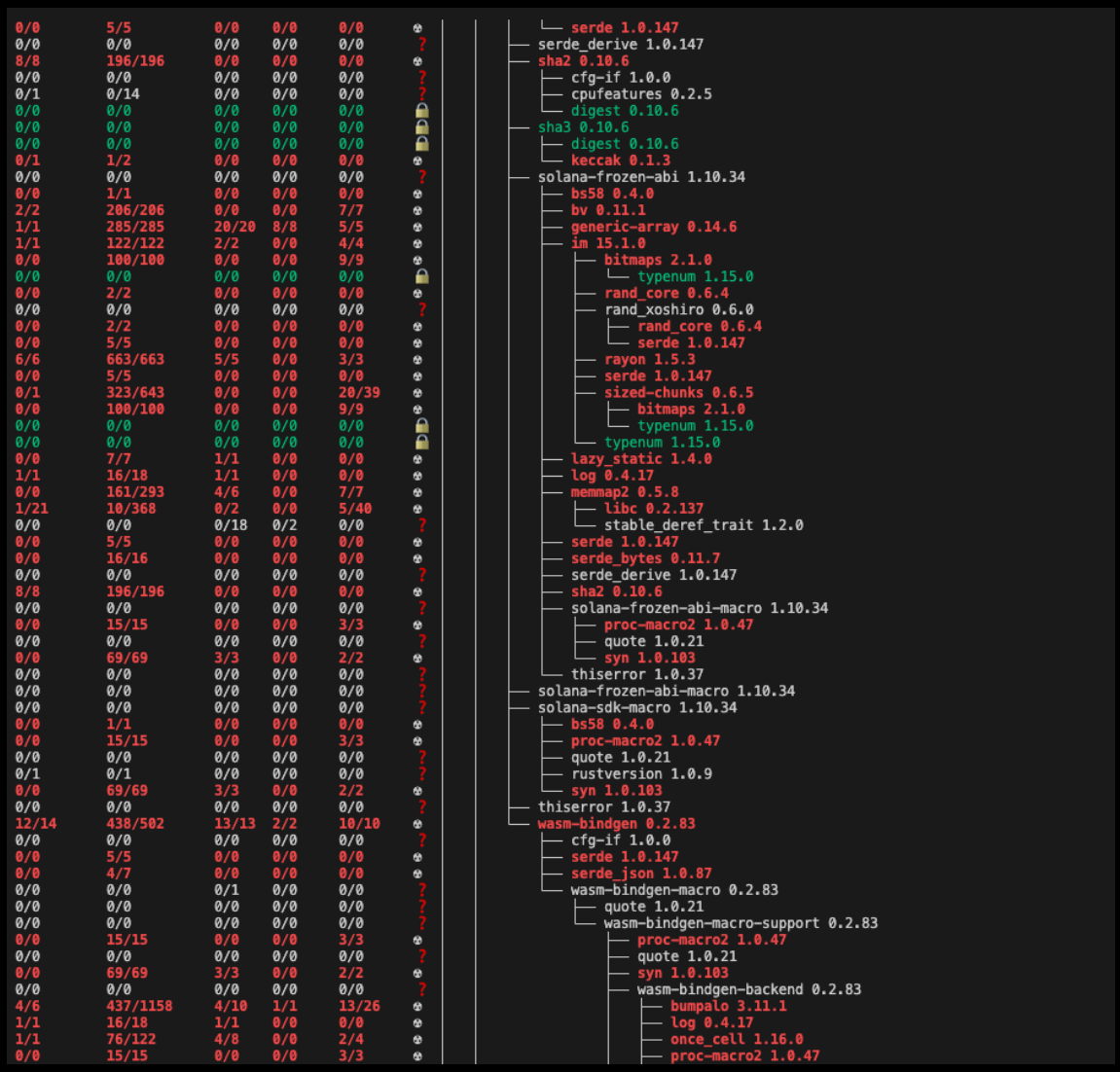


```
0/1       0/1         0/0    0/0    0/0      ?  │  │  ├── rustversion 1.0.9
0/0       69/69       3/3    0/0    2/2      ☢  │  │  └── syn 1.0.103
0/0       0/0         0/0    0/0    0/0      ?  │  ├── anchor-attribute-constant 0.25.0
0/0       8/8         0/0    0/0    0/0      ☢  │  │  ├── anchor-syn 0.25.0
0/0       15/15       0/0    0/0    3/3      ☢  │  │  ├── proc-macro2 1.0.47
0/0       69/69       3/3    0/0    2/2      ☢  │  │  └── syn 1.0.103
0/0       0/0         0/0    0/0    0/0      ?  │  ├── anchor-attribute-error 0.25.0
0/0       8/8         0/0    0/0    0/0      ☢  │  │  ├── anchor-syn 0.25.0
0/0       15/15       0/0    0/0    3/3      ☢  │  │  ├── proc-macro2 1.0.47
0/0       0/0         0/0    0/0    0/0      ?  │  │  ├── quote 1.0.21
0/0       69/69       3/3    0/0    2/2      ☢  │  │  └── syn 1.0.103
0/0       0/0         0/0    0/0    0/0      ?  │  ├── anchor-attribute-event 0.25.0
0/0       8/8         0/0    0/0    0/0      ☢  │  │  ├── anchor-syn 0.25.0
15/18     453/460     3/3    0/0    12/12    ☢  │  │  ├── anyhow 1.0.66
0/0       15/15       0/0    0/0    3/3      ☢  │  │  ├── proc-macro2 1.0.47
0/0       0/0         0/0    0/0    0/0      ?  │  │  ├── quote 1.0.21
0/0       69/69       3/3    0/0    2/2      ☢  │  │  └── syn 1.0.103
0/0       0/0         0/0    0/0    0/0      ?  │  ├── anchor-attribute-interface 0.25.0
0/0       8/8         0/0    0/0    0/0      ☢  │  │  ├── anchor-syn 0.25.0
15/18     453/460     3/3    0/0    12/12    ☢  │  │  ├── anyhow 1.0.66
0/0       0/0         0/0    0/0    0/0      ?  │  │  ├── heck 0.3.3
0/0       15/15       0/0    0/0    3/3      ☢  │  │  ├── proc-macro2 1.0.47
0/0       0/0         0/0    0/0    0/0      ?  │  │  ├── quote 1.0.21
0/0       69/69       3/3    0/0    2/2      ☢  │  │  └── syn 1.0.103
0/0       0/0         0/0    0/0    0/0      ?  │  ├── anchor-attribute-program 0.25.0
0/0       8/8         0/0    0/0    0/0      ☢  │  │  ├── anchor-syn 0.25.0
15/18     453/460     3/3    0/0    12/12    ☢  │  │  ├── anyhow 1.0.66
0/0       15/15       0/0    0/0    3/3      ☢  │  │  ├── proc-macro2 1.0.47
0/0       0/0         0/0    0/0    0/0      ?  │  │  ├── quote 1.0.21
0/0       69/69       3/3    0/0    2/2      ☢  │  │  └── syn 1.0.103
0/0       0/0         0/0    0/0    0/0      ?  │  ├── anchor-attribute-state 0.25.0
0/0       8/8         0/0    0/0    0/0      ☢  │  │  ├── anchor-syn 0.25.0
15/18     453/460     3/3    0/0    12/12    ☢  │  │  ├── anyhow 1.0.66
0/0       15/15       0/0    0/0    3/3      ☢  │  │  ├── proc-macro2 1.0.47
0/0       0/0         0/0    0/0    0/0      ?  │  │  ├── quote 1.0.21
0/0       69/69       3/3    0/0    2/2      ☢  │  │  └── syn 1.0.103
0/0       0/0         0/0    0/0    0/0      ?  │  ├── anchor-derive-accounts 0.25.0
0/0       8/8         0/0    0/0    0/0      ☢  │  │  ├── anchor-syn 0.25.0
15/18     453/460     3/3    0/0    12/12    ☢  │  │  ├── anyhow 1.0.66
0/0       15/15       0/0    0/0    3/3      ☢  │  │  ├── proc-macro2 1.0.47
0/0       0/0         0/0    0/0    0/0      ?  │  │  ├── quote 1.0.21
0/0       69/69       3/3    0/0    2/2      ☢  │  │  └── syn 1.0.103
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 0.13.1
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 1.0.147
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       15/15       0/0    0/0    3/3      ☢  │  │  │  │  ├── proc-macro2 1.0.47
0/0       0/0         0/0    0/0    0/0      ?  │  │  │  │  ├── quote 1.0.21
0/0       69/69       3/3    0/0    2/2      ☢  │  │  │  │  └── syn 1.0.103
0/0       0/0         0/0    0/0    0/0      ?  │  │  │  ├── borsh-schema-derive-internal 0.9.3
0/0       15/15       0/0    0/0    3/3      ☢  │  │  │  │  ├── proc-macro2 1.0.47
0/0       0/0         0/0    0/0    0/0      ?  │  │  │  │  ├── quote 1.0.21
0/0       69/69       3/3    0/0    2/2      ☢  │  │  │  │  └── syn 1.0.103
0/0       0/0         0/0    0/0    0/0      ?  │  │  │  ├── proc-macro-crate 0.1.5
0/0       0/0         0/0    0/0    0/0      🔒 │  │  │  │  └── toml 0.5.9
0/0       0/46        0/1    0/0    0/0      ?  │  │  │  │  ├── indexmap 1.9.1
0/0       5/5         0/0    0/0    0/0      ☢  │  │  │  │  └── serde 1.0.147
0/0       15/15       0/0    0/0    3/3      ☢  │  │  │  ├── proc-macro2 1.0.47
0/0       69/69       3/3    0/0    2/2      ☢  │  │  │  └── syn 1.0.103
2/2       1082/1198   19/22  1/1    51/58    ☢  │  │  └── hashbrown 0.11.2
```

AUTOMATED TESTING

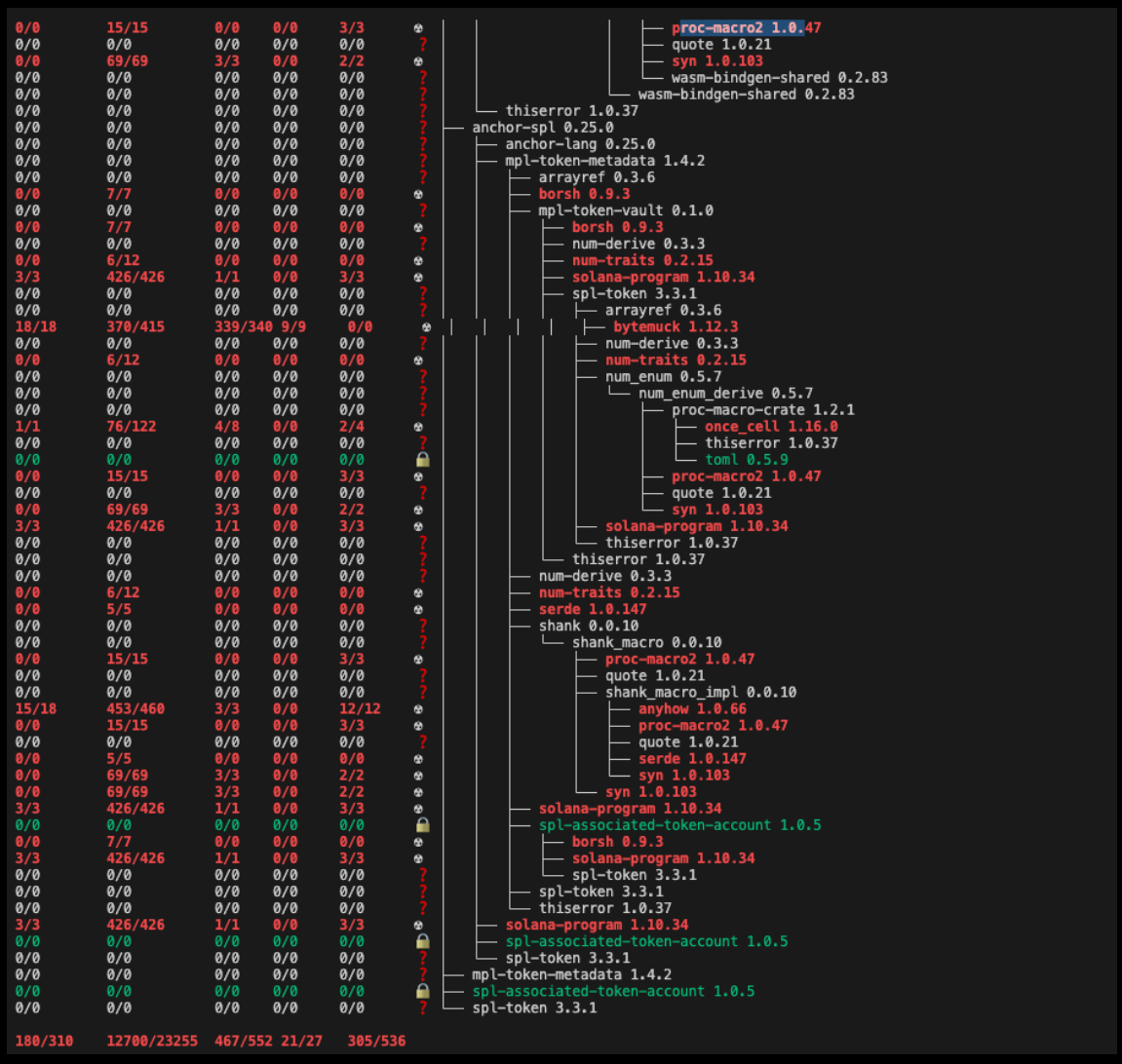


AUTOMATED TESTING



```
0/0     5/5     0/0   0/0   0/0     ☢
0/0     0/0     0/0   0/0   0/0     ?
8/8     196/196 0/0   0/0   0/0     ☢
0/0     0/0     0/0   0/0   0/0     ?
0/1     0/14    0/0   0/0   0/0     ?
0/0     0/0     0/0   0/0   0/0     🔒
0/0     0/0     0/0   0/0   0/0     🔒
0/0     0/0     0/0   0/0   0/0     🔒
0/1     1/2     0/0   0/0   0/0     ☢
0/0     0/0     0/0   0/0   0/0     ?
0/0     1/1     0/0   0/0   0/0     ☢
2/2     206/206 0/0   0/0   7/7     ☢
1/1     285/285 20/20 8/8   5/5     ☢
1/1     122/122 2/2   0/0   4/4     ☢
0/0     100/100 0/0   0/0   9/9     ☢
0/0     0/0     0/0   0/0   0/0     🔒
0/0     2/2     0/0   0/0   0/0     ☢
0/0     0/0     0/0   0/0   0/0     ?
0/0     2/2     0/0   0/0   0/0     ☢
0/0     5/5     0/0   0/0   0/0     ☢
6/6     663/663 5/5   0/0   3/3     ☢
0/0     5/5     0/0   0/0   0/0     ☢
0/1     323/643 0/0   0/0   20/39   ☢
0/0     100/100 0/0   0/0   9/9     ☢
0/0     0/0     0/0   0/0   0/0     🔒
0/0     0/0     0/0   0/0   0/0     🔒
0/0     7/7     1/1   0/0   0/0     ☢
1/1     16/18   1/1   0/0   0/0     ☢
0/0     161/293 4/6   0/0   7/7     ☢
1/21    10/368  0/2   0/0   5/40    ☢
0/0     0/0     0/18  0/2   0/0     ?
0/0     5/5     0/0   0/0   0/0     ☢
0/0     16/16   0/0   0/0   0/0     ☢
0/0     0/0     0/0   0/0   0/0     ?
8/8     196/196 0/0   0/0   0/0     ☢
0/0     0/0     0/0   0/0   0/0     ?
0/0     15/15   0/0   0/0   3/3     ☢
0/0     0/0     0/0   0/0   0/0     ?
0/0     69/69   3/3   0/0   2/2     ☢
0/0     0/0     0/0   0/0   0/0     ?
0/0     0/0     0/0   0/0   0/0     ?
0/0     0/0     0/0   0/0   0/0     ?
0/0     1/1     0/0   0/0   0/0     ☢
0/0     15/15   0/0   0/0   3/3     ☢
0/0     0/0     0/0   0/0   0/0     ?
0/1     0/1     0/0   0/0   0/0     ?
0/0     69/69   3/3   0/0   2/2     ☢
0/0     0/0     0/0   0/0   0/0     ?
12/14   438/502 13/13 2/2   10/10   ☢
0/0     0/0     0/0   0/0   0/0     ?
0/0     5/5     0/0   0/0   0/0     ☢
0/0     4/7     0/0   0/0   0/0     ☢
0/0     0/0     0/1   0/0   0/0     ?
0/0     0/0     0/0   0/0   0/0     ?
0/0     0/0     0/0   0/0   0/0     ?
0/0     15/15   0/0   0/0   3/3     ☢
0/0     0/0     0/0   0/0   0/0     ?
0/0     69/69   3/3   0/0   2/2     ☢
0/0     0/0     0/0   0/0   0/0     ?
4/6     437/1158 4/10 1/1   13/26   ☢
1/1     16/18   1/1   0/0   0/0     ☢
1/1     76/122  4/8   0/0   2/4     ☢
0/0     15/15   0/0   0/0   3/3     ☢

│   │   │   └── serde 1.0.147
│   │   ├── serde_derive 1.0.147
│   │   ├── sha2 0.10.6
│   │   │   ├── cfg-if 1.0.0
│   │   │   ├── cpufeatures 0.2.5
│   │   │   └── digest 0.10.6
│   │   ├── sha3 0.10.6
│   │   │   ├── digest 0.10.6
│   │   │   └── keccak 0.1.3
│   │   ├── solana-frozen-abi 1.10.34
│   │   │   ├── bs58 0.4.0
│   │   │   ├── bv 0.11.1
│   │   │   ├── generic-array 0.14.6
│   │   │   ├── im 15.1.0
│   │   │   │   ├── bitmaps 2.1.0
│   │   │   │   │   └── typenum 1.15.0
│   │   │   │   ├── rand_core 0.6.4
│   │   │   │   ├── rand_xoshiro 0.6.0
│   │   │   │   │   ├── rand_core 0.6.4
│   │   │   │   │   └── serde 1.0.147
│   │   │   │   ├── rayon 1.5.3
│   │   │   │   ├── serde 1.0.147
│   │   │   │   ├── sized-chunks 0.6.5
│   │   │   │   │   ├── bitmaps 2.1.0
│   │   │   │   │   └── typenum 1.15.0
│   │   │   │   └── typenum 1.15.0
│   │   │   ├── lazy_static 1.4.0
│   │   │   ├── log 0.4.17
│   │   │   ├── memmap2 0.5.8
│   │   │   │   ├── libc 0.2.137
│   │   │   │   └── stable_deref_trait 1.2.0
│   │   │   ├── serde 1.0.147
│   │   │   ├── serde_bytes 0.11.7
│   │   │   ├── serde_derive 1.0.147
│   │   │   ├── sha2 0.10.6
│   │   │   ├── solana-frozen-abi-macro 1.10.34
│   │   │   │   ├── proc-macro2 1.0.47
│   │   │   │   ├── quote 1.0.21
│   │   │   │   └── syn 1.0.103
│   │   │   └── thiserror 1.0.37
│   │   ├── solana-frozen-abi-macro 1.10.34
│   │   ├── solana-sdk-macro 1.10.34
│   │   │   ├── bs58 0.4.0
│   │   │   ├── proc-macro2 1.0.47
│   │   │   ├── quote 1.0.21
│   │   │   ├── rustversion 1.0.9
│   │   │   └── syn 1.0.103
│   │   ├── thiserror 1.0.37
│   │   └── wasm-bindgen 0.2.83
│   │       ├── cfg-if 1.0.0
│   │       ├── serde 1.0.147
│   │       ├── serde_json 1.0.87
│   │       └── wasm-bindgen-macro 0.2.83
│   │           ├── quote 1.0.21
│   │           └── wasm-bindgen-macro-support 0.2.83
│   │               ├── proc-macro2 1.0.47
│   │               ├── quote 1.0.21
│   │               ├── syn 1.0.103
│   │               ├── wasm-bindgen-backend 0.2.83
│   │               │   ├── bumpalo 3.11.1
│   │               │   ├── log 0.4.17
│   │               │   ├── once_cell 1.16.0
│   │               │   ├── proc-macro2 1.0.47
```

AUTOMATED TESTING



THANK YOU FOR CHOOSING

// HALBORN