HALBORN

Solana Labs - Perpetuals

Solana Program Security Audit

Prepared by: **Halborn**

Date of Engagement: **February 14th, 2023 - April 3rd, 2023**

Visit: **Halborn.com**

DOCUMENT REVISION HISTORY

VERSION	MODIFICATION	DATE	AUTHOR
0.1	Document Creation	02/23/2023	Isabel Burruezo
0.2	Document Updates	03/28/2023	Isabel Burruezo
0.3	Final Draft	03/31/2023	Isabel Burruezo
0.4	Draft Review	04/03/2023	Piotr Cielas
0.5	Draft Review	04/03/2023	Gabi Urrutia
1.0	Remediation Plan	04/20/2023	Isabel Burruezo
1.1	Remediation Plan Review	04/20/2023	Piotr Cielas
1.2	Remediation Plan Review	04/21/2023	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

EXECUTIVE OVERVIEW

1.1 INTRODUCTION

The Solana Perpetuals protocol is an open-source implementation of a noncustodial decentralized exchange that supports leveraged trading in a variety of assets.

Solana Labs engaged Halborn to conduct a security audit on their Solana program, beginning on February 14th, 2023 and ending on April 3rd, 2023 . The security assessment was scoped to the program provided in the perpetuals 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 seven 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 risks. The issue with higher security risk has been successfully addressed by Solana Labs , which is the following:

- Protocol fees and SOL fees locked permanently

Solana Labs acknowledged and accepted the risk of the rest of the findings since their impact were low and informational. In addition, some of them were confirmed that this is a feature and that this is expected behavior.

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 Solana 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.
- Program manual 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 lead to arithmetic vulnerabilities.
- Finding unsafe Rust code usage (cargo-geiger)
- Scanning dependencies for known vulnerabilities (cargo audit).
- Local runtime 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. Perpetuals

- Repository: perpetuals
- Commit ID: dc5b9076db580828dbd4d0291940c72694edb03d
- Programs in scope:
    1. perpetuals (perpetuals/program)

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

2. ASSESSMENT SUMMARY & FINDINGS OVERVIEW

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL
0	1	0	4	4

LIKELIHOOD

IMPACT

	(HAL-02) (HAL-03)			(HAL-01)
	(HAL-04) (HAL-05)			
(HAL-06) (HAL-07) (HAL-08) (HAL-09)				

EXECUTIVE OVERVIEW

SECURITY ANALYSIS	RISK LEVEL	REMEDIATION DATE
(HAL-01) PROTOCOL FEES AND SOL FEES LOCKED PERMANENTLY	High	SOLVED - 04/07/2023
(HAL-02) MINIMUM MULTISIG THRESHOLD CHECK MISSING	Low	RISK ACCEPTED
(HAL-03) ORACLE ADDRESSES CHECK MISSING	Low	RISK ACCEPTED
(HAL-04) CUSTODY TOKEN MINT ACCOUNT CHECK MISSING	Low	RISK ACCEPTED
(HAL-05) CUSTODY CONFIG VALUES CAN BE UPDATED ANYTIME	Low	RISK ACCEPTED
(HAL-06) PERPETUAL PERMISSIONS ARE APPLIED GLOBALLY	Informational	ACKNOWLEDGED
(HAL-07) REDUNDANT FIELDS OF THE PERPETUALS ACCOUNT	Informational	ACKNOWLEDGED
(HAL-08) REDUNDANT FUNCTION	Informational	ACKNOWLEDGED
(HAL-09) POSSIBLE RUST PANICS DUE TO UNSAFE UNWRAP USAGE	Informational	ACKNOWLEDGED

FINDINGS & TECH DETAILS

3.1 (HAL-01) PROTOCOL FEES AND SOL FEES LOCKED PERMANENTLY - HIGH

Description:

The WithdrawFees instruction allows admins to transfer protocol fees from the custody_token_account recollected and SOL fees. Those SOL fees are the fees collected when some accounts are removed like the custody or pool and the rent is transferred to the transfer_authority.

To withdraw SOL it is necessary:
* it is necessary the transfer authority's lamport balance is greater than the rent-exempt minimum. This happens when the custody account is removed, otherwise the instruction fails. However, to remove the custody account , however it is necessary the balance of the custody_token_account is zero.

To withdraw protocol fees:
* it is necessary to provide the custody account and the custody_token_account. The former account then must exist, and the second one needs to hold more tokens than the amount required to withdraw.

As can be seen in the above description, the requirements for withdrawing the fees conflict so that the instruction always fails and it is not possible to withdraw any of protocol fees or SOL fees.

Code Location:

Listing 1: src/instructions/withdraw_fees.rs

```
120  msg!(
121              "Withdraw token fees: {} / {}",
122              params.token_amount,
123              custody.assets.protocol_fees
124          );
```

```
125
126         if custody.assets.protocol_fees < params.token_amount {
127             return Err(ProgramError::InsufficientFunds.into());
128         }
129         custody.assets.protocol_fees =
130             math::checked_sub(custody.assets.protocol_fees, params
 ↳ .token_amount)?;
131
132         ctx.accounts.perpetuals.transfer_tokens(
133             ctx.accounts.custody_token_account.to_account_info(),
134             ctx.accounts.receiving_token_account.to_account_info()
 ↳ ,
135             ctx.accounts.transfer_authority.to_account_info(),
136             ctx.accounts.token_program.to_account_info(),
137             params.token_amount,
138         )?;
139     }
140
141     // transfer sol fees from the custody to the receiver
142     if params.sol_amount > 0 {
143         let balance = ctx.accounts.transfer_authority.try_lamports
 ↳ ()?;
144         let min_balance = sysvar::rent::Rent::get().unwrap().
 ↳ minimum_balance(0);
145
146         let available_balance = if balance > min_balance {
147             math::checked_sub(balance, min_balance)?
148         } else {
149             0
150         };
151
152         msg!(
153             "Withdraw SOL fees: {} / {}",
154             params.sol_amount,
155             available_balance
156         );
157
158         if available_balance < params.sol_amount {
159             return Err(ProgramError::InsufficientFunds.into());
160         }
161
162         Perpetuals::transfer_sol_from_owned(
163             ctx.accounts.transfer_authority.to_account_info(),
164             ctx.accounts.receiving_sol_account.to_account_info(),
```

```
165             params.sol_amount,
166         )?;
```

Listing 2: src/instructions/remove_custody.rs (Lines 101,111)

```
100  require!(
101          ctx.accounts.custody_token_account.amount == 0,
102          PerpetualsError::InvalidCustodyState
103      );
104
105      // remove token from the list
106      let pool = ctx.accounts.pool.as_mut();
107      let token_id = pool.get_token_id(&ctx.accounts.custody.key())
 ↳ ?;
108      pool.tokens.remove(token_id);
109
110      Perpetuals::close_token_account(
111          ctx.accounts.transfer_authority.to_account_info(),
112          ctx.accounts.custody_token_account.to_account_info(),
113          ctx.accounts.token_program.to_account_info(),
114          ctx.accounts.transfer_authority.to_account_info(),
115          &[&[
116              b"transfer_authority",
117              &[ctx.accounts.perpetuals.transfer_authority_bump],
118          ]],
119      )?;
```

Listing 3: src/instructions/remove_pool.rs (Lines 86,87)

```
 75  if signatures_left > 0 {
 76          msg!(
 77              "Instruction has been signed but more signatures are
 ↳ required: {}",
 78              signatures_left
 79          );
 80          msg!("signatures_left: {:?}", signatures_left);
 81
 82          return Ok(signatures_left);
 83      }
 84
 85      require!(
 86          ctx.accounts.pool.tokens.is_empty(),
 87          PerpetualsError::InvalidPoolState
```

```
88      );
89
90      // remove pool from the list
91      let perpetuals = ctx.accounts.perpetuals.as_mut();
92      let pool_idx = perpetuals
93          .pools
94          .iter()
95          .position(|x| *x == ctx.accounts.pool.key())
96          .ok_or::<Error>(PerpetualsError::InvalidPoolState.into())
↳ ?;
97      perpetuals.pools.remove(pool_idx);
```

Risk Level:

Likelihood - 5
Impact - 3

Proof of Concept:

Steps To Reproduce

1) Init Perpetuals
2) Add Pool
3) Add Custody
4) Alice adds Liquidity
5) Alice opens a position
6) Withdraw Fees

Notice that although it would be possible to withdraw the protocol fees, the SOL fees cannot be withdrawn since the position is not closed.

```
[**]WithdrawFees Intruction!
Receiving  token account : 1000000
Custody  token account : 1040802
[+] custody assets's  Account data:  --->  Assets { collateral_usd: 360000000, protocol_fees: 4, owned: 398, locked: 20 }
[2023-04-03T13:57:08.329716000Z DEBUG solana_runtime::message_processor::stable_log] Program PERP9EeXeGnyEqGmxQSan4nGRAFNLwTufLJmiYsTJ8j invoke [1]
[2023-04-03T13:57:08.331072000Z DEBUG solana_runtime::message_processor::stable_log] Program log: Instruction: WithdrawFees
[2023-04-03T13:57:08.341774000Z DEBUG solana_runtime::message_processor::stable_log] Program log: Withdraw token fees: 2 / 4
[2023-04-03T13:57:08.355738000Z DEBUG solana_runtime::message_processor::stable_log] Program TokenkegQfeZyiNwAJbNbGKPFXCWuBvf9Ss623VQ5DA invoke [2]
[2023-04-03T13:57:08.356449000Z DEBUG solana_runtime::message_processor::stable_log] Program log: Instruction: Transfer
[2023-04-03T13:57:08.359063000Z DEBUG solana_runtime::message_processor::stable_log] Program TokenkegQfeZyiNwAJbNbGKPFXCWuBvf9Ss623VQ5DA consumed 4880 of 171950 compute units
[2023-04-03T13:57:08.359177000Z DEBUG solana_runtime::message_processor::stable_log] Program TokenkegQfeZyiNwAJbNbGKPFXCWuBvf9Ss623VQ5DA success
[2023-04-03T13:57:08.360164000Z DEBUG solana_runtime::message_processor::stable_log] Program log: balance: 890880
[2023-04-03T13:57:08.360558000Z DEBUG solana_runtime::message_processor::stable_log] Program log: min_balance: 890880
[2023-04-03T13:57:08.361028000Z DEBUG solana_runtime::message_processor::stable_log] Program log: Withdraw SOL fees: 2 / 0
[2023-04-03T13:57:08.362293000Z DEBUG solana_runtime::message_processor::stable_log] Program log: ProgramError occurred. Error Code: InsufficientFunds. Error Number: 25769803776. Error Message: An account
's balance was too small to complete the instruction.
[2023-04-03T13:57:08.362805000Z DEBUG solana_runtime::message_processor::stable_log] Program PERP9EeXeGnyEqGmxQSan4nGRAFNLwTufLJmiYsTJ8j consumed 38013 of 200000 compute units
[2023-04-03T13:57:08.362893000Z DEBUG solana_runtime::message_processor::stable_log] Program PERP9EeXeGnyEqGmxQSan4nGRAFNLwTufLJmiYsTJ8j failed: insufficient funds for instruction
thread 'poc_liquidity' panicked at 'called `Result::unwrap()` on an `Err` value: TransactionError(InstructionError(0, InsufficientFunds))', programs/perpetuals/tests/security.rs:2468:45
note: run with `RUST_BACKTRACE=1` environment variable to display a backtrace
```

Recommendation:

It is recommended to replace the WithdrawFees instruction with two independent instructions to withdraw the protocol fees at any time, regardless of the custody account. This way, it is possible to withdraw SOL fees when custody is closed without interfering in withdrawing protocol fees.

Remediation Plan:

SOLVED: The Solana Labs team fixed this issue in commit:
- 84bb60bec61b5a463c506f0535567d00f9e59b21:
The WithdrawFees instruction has been split in two separate instructions, WithdrawFees and WithdrawSOLFees to withdraw **protocol fees** and **SOL fees** respectively. This way it is possible to withdraw both types of fees independently and successfully fulfilling the necessary requirements for both.

3.2 (HAL-02) MINIMUM MULTISIG THRESHOLD CHECK MISSING - LOW

Description:

The Init instruction allows the upgrade authority to initialize the perpetuals account and set the multisig's signers. The instruction handler requires the transaction sender to provide a selection of accounts and parameters, including min_signatures. This parameter sets the signatures threshold required for a transaction to be considered valid. Likewise, the SetAdminSigners instruction handler allows setting a new signers list and new min_signatures.

However, both instructions handlers allow setting the min_signatures field of multisig with a value equal to 1. Setting the threshold to 1 results in having no multisig functionality at all because only one user controls the account.

Code Location:

Listing 4: src/instruction/init.rs (Lines 63,78)

```
62 pub struct InitParams {
63     pub min_signatures: u8,
64     pub allow_swap: bool,
65     pub allow_add_liquidity: bool,
66     pub allow_remove_liquidity: bool,
67     pub allow_open_position: bool,
68     pub allow_close_position: bool,
69     pub allow_pnl_withdrawal: bool,
70     pub allow_collateral_withdrawal: bool,
71     pub allow_size_change: bool,
72 }
73
74 pub fn init(ctx: Context<Init>, params: &InitParams) -> Result<()>
↳  {
75     // initialize multisig, this will fail if account is already
↳ initialized
76     let mut multisig = ctx.accounts.multisig.load_init()?;
```

```
77
78      multisig.set_signers(ctx.remaining_accounts, params.
↳ min_signatures)?;
```

Listing 5: src/state/multisig.rs (Line 16)

```
13 pub struct Multisig {
14     pub num_signers: u8,
15     pub num_signed: u8,
16     pub min_signatures: u8,
17     pub instruction_accounts_len: u8,
18     pub instruction_data_len: u16,
19     pub instruction_hash: u64,
20     pub signers: [Pubkey; 6], // Multisig::MAX_SIGNERS
21     pub signed: [bool; 6],    // Multisig::MAX_SIGNERS
22     pub bump: u8,
23 }
```

Listing 6: src/instruction/set_admin_signers.rs (Line 24)

```
23 pub struct SetAdminSignersParams {
24     pub min_signatures: u8,
25 }
26
27 pub fn set_admin_signers<'info>(
28     ctx: Context<'_, '_, '_, 'info, SetAdminSigners<'info>>,
29     params: &SetAdminSignersParams,
```

Risk Level:

Likelihood - 2
Impact - 3

Recommendation:

It is recommended to add a check to verify that the value of `min_signatures` passed as a parameter is equal to or greater than three, as well as the number of remaining accounts provided as admins .

Remediation Plan:

RISK ACCEPTED: The Solana Labs team accepted the risk of this finding.

3.3 (HAL-03) ORACLE ADDRESSES CHECK MISSING - LOW

Description:

The Oracle accounts required by the AddCustody and SetCustodyConfig instruction handlers are not validated. Although it is up to the administrators to provide the oracle accounts, if they mistakenly provide the wrong oracle account, the perpetuals program may end up using malicious price feeders.

Code Location:

Listing 7: src/instruction/add_custody.rs (Line 87)

```
85 pub struct AddCustodyParams {
86     pub is_stable: bool,
87     pub oracle: OracleParams,
88     pub pricing: PricingParams,
89     pub permissions: Permissions,
90     pub fees: Fees,
91     pub target_ratio: u64,
92     pub min_ratio: u64,
93     pub max_ratio: u64,
94 }
95
96 pub fn add_custody<'info>(
```

Listing 8: src/state/custody.rs (Line 147)

```
72 pub struct OracleParams {
73     pub oracle_account: Pubkey,
74     pub oracle_type: OracleType,
75     pub max_price_error: u64,
76     pub max_price_age_sec: u32,
77 }
```

Listing 9: src/instruction/add_custody.rs (Line 147)

```
140  let custody = ctx.accounts.custody.as_mut();
141
142      custody.pool = pool.key();
143      custody.mint = ctx.accounts.custody_token_mint.key();
144      custody.token_account = ctx.accounts.custody_token_account.key
 ↳ ();
145      custody.decimals = ctx.accounts.custody_token_mint.decimals;
146      custody.is_stable = params.is_stable;
147      custody.oracle = params.oracle;
148      custody.pricing = params.pricing;
149      custody.permissions = params.permissions;
150      custody.fees = params.fees;
151      custody.bump = *ctx.bumps.get("custody").ok_or(ProgramError::
 ↳ InvalidSeeds)?;
152      custody.token_account_bump = *ctx
153          .bumps
154          .get("custody_token_account")
155          .ok_or(ProgramError::InvalidSeeds)?;
156
157      if !custody.validate() {
158          err!(PerpetualsError::InvalidCustodyConfig)
159      } else {
160          Ok(0)
```

Risk Level:

Likelihood - 2
Impact - 3

Recommendation:

It is recommended to verify if the oracle account's owner matches a known and trusted address before.

Remediation Plan:

RISK ACCEPTED: The Solana Labs team accepted the risk of this finding.

3.4 (HAL-04) CUSTODY TOKEN MINT ACCOUNT CHECK MISSING - LOW

Description:

The AddCustody instruction handler requires the transaction sender to provide a selection of accounts, including the custody_token_mint account.

The decimals field of the mint account determines the custody.decimals value, which the instruction handlers used when calculating fees charged by the add_liquidity and add_collateral functions. However, this custody_token_mint provided account is not checked and neither are its fields, so the decimals value could be other than expected. This could result in a direct impact to fee calculation results.
It is important to mention that the freeze_authority field is not checked either.

Code Location:

Listing 10: src/instruction/add_custody.rs (Line 77)

```
53 #[account(
54     init_if_needed,
55     payer = admin,
56     space = Custody::LEN,
57     seeds = [b"custody",
58              pool.key().as_ref(),
59              custody_token_mint.key().as_ref()],
60     bump
61 )]
62 pub custody: Box<Account<'info, Custody>>,
63
64 #[account(
65     init_if_needed,
66     payer = admin,
67     token::mint = custody_token_mint,
68     token::authority = transfer_authority,
69     seeds = [b"custody_token_account",
```

```
70                 pool.key().as_ref(),
71                 custody_token_mint.key().as_ref()],
72         bump
73 )]
74 pub custody_token_account: Box<Account<'info, TokenAccount>>,
75
76 #[account()]
77 pub custody_token_mint: Box<Account<'info, Mint>>,
```

Listing 11: src/instruction/add_custody.rs (Lines 143,145)

```
140 let custody = ctx.accounts.custody.as_mut();
141
142 custody.pool = pool.key();
143 custody.mint = ctx.accounts.custody_token_mint.key();
144 custody.token_account = ctx.accounts.custody_token_account.key();
145 custody.decimals = ctx.accounts.custody_token_mint.decimals;
146 custody.is_stable = params.is_stable;
```

Listing 12: src/instruction/add_custody.rs (Line 203)

```
198 // update custody stats
199 msg!("Update custody stats");
200 custody.collected_fees.add_liquidity_usd = custody
201         .collected_fees
202         .add_liquidity_usd
203         .wrapping_add(token_price.get_asset_amount_usd(fee_amount,
 ↳ custody.decimals)?);
```

Risk Level:

Likelihood - 2
Impact - 2

Recommendation:

It is recommended to add a check to verify the mint's decimals and authorities are the expected and corresponding ones.

Remediation Plan:

RISK ACCEPTED: The Solana Labs team accepted the risk of this finding.

3.5 (HAL-05) CUSTODY CONFIG VALUES CAN BE UPDATED ANYTIME - LOW

Description:

The AddCustody instruction handler requires multiple parameters to add a custody, including **Fees** and **Permissions**.
Those custody parameters can be updated anytime and affect positions and deposits retroactively. The legacy parameter values are not preserved.

This happens in the same way with permission values, they can be changed at any time by the admin signers to allow or disallow to open or close positions, add and remove liquidity among others, as it is explained in HAL_06.

Code Location:

Listing 13: src/instructions/add_custody.rs (Line 90)

```
85 pub struct AddCustodyParams {
86     pub is_stable: bool,
87     pub oracle: OracleParams,
88     pub pricing: PricingParams,
89     pub permissions: Permissions,
90     pub fees: Fees,
91     pub target_ratio: u64,
92     pub min_ratio: u64,
93     pub max_ratio: u64,
94 }
95
96 pub fn add_custody<'info>(
97     ctx: Context<'_, '_, '_, 'info, AddCustody<'info>>,
98     params: &AddCustodyParams,
```

Listing 14: src/state/custody.rs

```
16 pub struct Fees {
17     pub mode: FeesMode,
18     // fees have implied BPS_DECIMALS decimals
```

```
19      pub max_increase: u64,
20      pub max_decrease: u64,
21      pub swap: u64,
22      pub add_liquidity: u64,
23      pub remove_liquidity: u64,
24      pub open_position: u64,
25      pub close_position: u64,
26      pub liquidation: u64,
27      pub protocol_share: u64,
28  }
```

Listing 15: src/instructions/set_custody_config.rs (Line 97)

```
47  pub struct SetCustodyConfigParams {
48      pub is_stable: bool,
49      pub oracle: OracleParams,
50      pub pricing: PricingParams,
51      pub permissions: Permissions,
52      pub fees: Fees,
53      pub target_ratio: u64,
54      pub min_ratio: u64,
55      pub max_ratio: u64,
56  }
57
58  pub fn set_custody_config<'info>(
59      ctx: Context<'_, '_, '_, 'info, SetCustodyConfig<'info>>,
60      params: &SetCustodyConfigParams,
61  ) -> Result<u8> {
62      // validate inputs
63      if params.min_ratio > params.target_ratio || params.
↳ target_ratio > params.max_ratio {
64          return Err(ProgramError::InvalidArgument.into());
65      }
66
67      // validate signatures
68      let mut multisig = ctx.accounts.multisig.load_mut()?;
69
70      let signatures_left = multisig.sign_multisig(
71          &ctx.accounts.admin,
72          &Multisig::get_account_infos(&ctx)[1..],
73          &Multisig::get_instruction_data(AdminInstruction::
↳ SetCustodyConfig, params)?,
74      )?;
75      if signatures_left > 0 {
```

```
76          msg!(
77              "Instruction has been signed but more signatures are
↳ required: {}",
78              signatures_left
79          );
80          return Ok(signatures_left);
81      }
82
83      // update pool data
84      let pool = ctx.accounts.pool.as_mut();
85      let idx = pool.get_token_id(&ctx.accounts.custody.key())?;
86      pool.tokens[idx].target_ratio = params.target_ratio;
87      pool.tokens[idx].min_ratio = params.min_ratio;
88      pool.tokens[idx].max_ratio = params.max_ratio;
89
90      // update custody data
91      let custody = ctx.accounts.custody.as_mut();
92      custody.is_stable = params.is_stable;
93      custody.oracle = params.oracle;
94      custody.pricing = params.pricing;
95      custody.permissions = params.permissions;
96      custody.fees = params.fees;
```

Risk Level:

Likelihood - 2
Impact - 2

Recommendation:

Consider adding a field in the position to preserve commission rates in order to keep the original ones for each of them at the time when they were opened.

Remediation Plan:

RISK ACCEPTED: The Solana Labs team accepted the risk of this finding.

3.6 (HAL-06) PERPETUAL PERMISSIONS ARE APPLIED GLOBALLY - INFORMATIONAL

Description:

The Init instruction allows the upgrade authority to initialize the perpetuals account with permissions and set the multisig's signers. In addition, the SetPermissions instruction allows changing those permissions for a perpetual's account.

The AddCustody and SetCustodyConfig instructions require some parameters like **Permissions** discussed in **HAL-05** which are used to initialize and set up the custody.

These permissions allow managing access to certain actions like adding and removing liquidity, opening and closing positions, withdrawing collateral, among others.

However, if any of the allow_collateral_withdrawal, allow_close_position or allow_remove_liquidity perpetuals account permissions are updated, it is not possible to carry out that action for any custody. Thus, the collateral and liquidity cannot be withdrawn by the owners because it is locked until unlocked. This could result in an issue in a scenario described in HAL-02.

Code Location:

Listing 16: src/state/perpetuals.rs

```
26 pub struct Permissions {
27     pub allow_swap: bool,
28     pub allow_add_liquidity: bool,
29     pub allow_remove_liquidity: bool,
30     pub allow_open_position: bool,
31     pub allow_close_position: bool,
32     pub allow_pnl_withdrawal: bool,
```

```
33      pub allow_collateral_withdrawal: bool,
34      pub allow_size_change: bool,
```

Listing 17: src/instructions/set_permissions.rs

```
26 pub struct SetPermissionsParams {
27      pub allow_swap: bool,
28      pub allow_add_liquidity: bool,
29      pub allow_remove_liquidity: bool,
30      pub allow_open_position: bool,
31      pub allow_close_position: bool,
32      pub allow_pnl_withdrawal: bool,
33      pub allow_collateral_withdrawal: bool,
34      pub allow_size_change: bool,
35 }
36
37 pub fn set_permissions<'info>(
38      ctx: Context<'_, '_, '_, 'info, SetPermissions<'info>>,
39      params: &SetPermissionsParams,
40 ) -> Result<u8> {
```

Listing 18: src/instructions/set_permissions.rs

```
69      // update permissions
70      let perpetuals = ctx.accounts.perpetuals.as_mut();
71      perpetuals.permissions.allow_swap = params.allow_swap;
72      perpetuals.permissions.allow_add_liquidity = params.
↳ allow_add_liquidity;
73      perpetuals.permissions.allow_remove_liquidity = params.
↳ allow_remove_liquidity;
74      perpetuals.permissions.allow_open_position = params.
↳ allow_open_position;
75      perpetuals.permissions.allow_close_position = params.
↳ allow_close_position;
76      perpetuals.permissions.allow_pnl_withdrawal = params.
↳ allow_pnl_withdrawal;
77      perpetuals.permissions.allow_collateral_withdrawal = params.
↳ allow_collateral_withdrawal;
78      perpetuals.permissions.allow_size_change = params.
↳ allow_size_change;
```

Listing 19: src/instructions/remove_collateral.rs (Line 105)

```
95 pub fn remove_collateral(
96     ctx: Context<RemoveCollateral>,
97     params: &RemoveCollateralParams,
98 ) -> Result<()> {
99     // check permissions
100     msg!("Check permissions");
101
102     let perpetuals = ctx.accounts.perpetuals.as_mut();
103     let custody = ctx.accounts.custody.as_mut();
104     require!(
105         perpetuals.permissions.allow_collateral_withdrawal
106             && custody.permissions.allow_collateral_withdrawal,
107         PerpetualsError::InstructionNotAllowed
108     );
```

Risk Level:

Likelihood - 1
Impact - 1

Recommendation:

It is recommended to carry out the recommendation plan for HAL-02 to make sure the multisig is activated.

Remediation Plan:

ACKNOWLEDGED: The Solana Labs team acknowledged this finding.

3.7 (HAL-07) REDUNDANT FIELDS OF THE PERPETUALS ACCOUNT - INFORMATIONAL

Description:

The Init instruction allows initializing the perpetuals account providing some permissions field's values, also required by the SetPermissions, AddCustody and SetCustodyConfig instructions. They are used to setting permissions in the program and the custody accounts to allow some operations to be carried out or not. However, allow_pnl_withdrawal and allow_size_change, are never used in the program.

Code Location:

Listing 20: src/state/perpetuals.rs (Lines 32,34)

```
26 pub struct Permissions {
27     pub allow_swap: bool,
28     pub allow_add_liquidity: bool,
29     pub allow_remove_liquidity: bool,
30     pub allow_open_position: bool,
31     pub allow_close_position: bool,
32     pub allow_pnl_withdrawal: bool,
33     pub allow_collateral_withdrawal: bool,
34     pub allow_size_change: bool,
35 }
```

Listing 21: src/instructions/init.rs (Lines 69,71)

```
62 pub struct InitParams {
63     pub min_signatures: u8,
64     pub allow_swap: bool,
65     pub allow_add_liquidity: bool,
66     pub allow_remove_liquidity: bool,
67     pub allow_open_position: bool,
68     pub allow_close_position: bool,
69     pub allow_pnl_withdrawal: bool,
70     pub allow_collateral_withdrawal: bool,
71     pub allow_size_change: bool,
```

```
72 }
73
74 pub fn init(ctx: Context<Init>, params: &InitParams) -> Result<()>
↳  {
```

Listing 22: src/instructions/set_permissions.rs (Lines 41,43)

```
35 pub struct SetPermissionsParams {
36     pub allow_swap: bool,
37     pub allow_add_liquidity: bool,
38     pub allow_remove_liquidity: bool,
39     pub allow_open_position: bool,
40     pub allow_close_position: bool,
41     pub allow_pnl_withdrawal: bool,
42     pub allow_collateral_withdrawal: bool,
43     pub allow_size_change: bool,
44 }
45
46 pub fn set_permissions<'info>(
47     ctx: Context<'_, '_, '_, 'info, SetPermissions<'info>>,
48     params: &SetPermissionsParams,
```

Listing 23: src/instructions/add_custody.rs (Line 89)

```
86     pub is_stable: bool,
87     pub oracle: OracleParams,
88     pub pricing: PricingParams,
89     pub permissions: Permissions,
90     pub fees: Fees,
91     pub target_ratio: u64,
92     pub min_ratio: u64,
93     pub max_ratio: u64,
94 }
95
96 pub fn add_custody<'info>(
97     ctx: Context<'_, '_, '_, 'info, AddCustody<'info>>,
98     params: &AddCustodyParams,
```

Risk Level:

Likelihood - 1
Impact - 1

Recommendation:

It is recommended to complete the implementation to make use of these fields or remove them.

Remediation Plan:

ACKNOWLEDGED: The Solana Labs team acknowledged this finding.

3.8 (HAL-08) REDUNDANT FUNCTION - INFORMATIONAL

Description:

The unsign_multisig() function allows removing the admin signature from the multisig. However, it has been detected that this function is not used in the program.

Code Location:

Listing 24: src/state/multisig.rs

```
200
201  pub fn unsign_multisig(&mut self, signer_account: &AccountInfo)
 ↳ -> Result<()> {
202          // return early if not a signer
203          if !signer_account.is_signer {
204              return Err(ProgramError::MissingRequiredSignature.into
 ↳ ());
205          }
206
207          // if single signer return
208          if self.num_signers <= 1 || self.num_signed == 0 {
209              return Ok(());
210          }
211
212          // find index of current signer or return error if not
 ↳ found
213          let signer_idx = if let Ok(idx) = self.get_signer_index(
 ↳ signer_account.key) {
214              idx
215          } else {
216              return err!(PerpetualsError::
 ↳ MultisigAccountNotAuthorized);
217          };
218
219          // if not signed by this account return
220          if !self.signed[signer_idx] {
221              return Ok(());
222          }
```

```
223
224         // remove signature
225         self.num_signed -= 1;
226         self.signed[signer_idx] = false;
227
228         Ok(())
229     }
```

Risk Level:

Likelihood - 1
Impact - 1

Recommendation:

It is recommended to complete the implementation to make use of this function or remove it.

Remediation Plan:

ACKNOWLEDGED: The Solana Labs team acknowledged this finding.

3.9 (HAL-09) POSSIBLE RUST PANICS DUE TO UNSAFE UNWRAP USAGE - INFORMATIONAL

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 the 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:

Listing 25

```
./instructions/withdraw_fees.rs:144:        let min_balance =
↳ sysvar::rent::Rent::get().unwrap().minimum_balance(0);
./state/oracle.rs:321:          assert_eq!(12.3, price.
↳ checked_as_f64().unwrap());
./state/oracle.rs:324:          assert_eq!(12300000.0, price.
↳ checked_as_f64().unwrap());
./state/oracle.rs:330:          let scaled = price.scale_to_exponent
↳ (-6).unwrap();
./state/oracle.rs:334:          let scaled = price.scale_to_exponent
↳ (-1).unwrap();
./state/oracle.rs:338:          let scaled = price.scale_to_exponent
↳ (1).unwrap();
./state/pool.rs:894:          math::checked_mul(amount, 10u64.pow(
↳ decimals as u32)).unwrap()
./state/pool.rs:899:            math::checked_float_mul(amount, 10
↳ u64.pow(decimals as u32) as f64).unwrap(),
./state/pool.rs:901:          .unwrap()
./state/pool.rs:911:                  .unwrap()
./state/pool.rs:917:                  .unwrap()
./state/pool.rs:924:          assert_eq!(0, pool.get_new_ratio(0, 0,
↳  &custody, &token_price).unwrap());
./state/pool.rs:932:                  .unwrap()
```

```
./state/pool.rs:938:                .unwrap()
./state/pool.rs:944:                .unwrap()
./state/pool.rs:949:            pool.get_new_ratio(0, 0, &custody,
↳  &token_price).unwrap()
./state/pool.rs:968:            .unwrap()
./state/pool.rs:982:            .unwrap()
./state/pool.rs:1001:            .unwrap()
./state/pool.rs:1015:            .unwrap()
./state/pool.rs:1029:            .unwrap()
./state/pool.rs:1045:            .unwrap()
./state/pool.rs:1058:            .unwrap()
./state/pool.rs:1069:                .unwrap()
./state/pool.rs:1076:                .unwrap()
./state/pool.rs:1083:                .unwrap()
./state/pool.rs:1094:                .unwrap()
./state/pool.rs:1101:                .unwrap()
./state/pool.rs:1108:                .unwrap()
./state/pool.rs:1115:                .unwrap()
./state/pool.rs:1122:                .unwrap()
./state/pool.rs:1129:                .unwrap()
./state/pool.rs:1136:                .unwrap()
./state/pool.rs:1147:                .unwrap()
./state/pool.rs:1154:                .unwrap()
./state/pool.rs:1161:                .unwrap()
./state/pool.rs:1168:                .unwrap()
./state/pool.rs:1175:                .unwrap()
./state/pool.rs:1182:                .unwrap()
./state/pool.rs:1205:            .unwrap()
```

Risk Level:

Likelihood - 1
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:

ACKNOWLEDGED: The Solana Labs team acknowledged this finding.

MANUAL TESTING

In the manual testing phase, the following scenarios were simulated. The scenarios listed below were selected based on the severity of the vulnerabilities Halborn was testing the program for.

4.1 CLOSE CUSTODY WHEN CUSTODY TOKEN ACCOUNT AMOUNT IS NOT ZERO YET

Description:

The `RemoveCustody` instruction allows closing the custody account.
To achieve this, it removes the pool from the token list of the perpetuals account and closes the custody account, transferring the remaining lamports to the `transfer_authority` account.

It has been tested whether this instruction can be called at any time, so that `custody_token_account` tokens would no longer be available for all other operations.

Results:

No vulnerabilities were identified.

4.2 REMOVE POOL BEFORE CUSTODY ACCOUNT

Description:

The RemovePool and RemoveCustody instructions allow closing pool and custody accounts, respectively. A custody account has a field for the pool account associated to it. It has been tested if it is possible to remove a pool before the custody to check if there could be an inconsistency that could result in a vulnerability.

Results:

No vulnerabilities were identified.

```
[**]AddCustody Intruction done!
[**]RemovePool Intruction!
[2023-02-27T06:36:06.094570000Z DEBUG solana_runtime::message_processor::stable_log] Program PERP9EeXeGnyEqGmxGSan4nGRAFNLwTufLJmiYsTJ8j invoke [1]
[2023-02-27T06:36:06.095493000Z DEBUG solana_runtime::message_processor::stable_log] Program log: Instruction: RemovePool
[2023-02-27T06:36:06.102311000Z DEBUG solana_runtime::message_processor::stable_log] Program log: pool.tokens is empty?: false
[2023-02-27T06:36:06.103933000Z DEBUG solana_runtime::message_processor::stable_log] Program log: AnchorError thrown in programs/perpetuals/src/instructions/remove_pool
.rs:84. Error Code: InvalidPoolState. Error Number: 6010. Error Message: Invalid pool state.
[2023-02-27T06:36:06.104342000Z DEBUG solana_runtime::message_processor::stable_log] Program PERP9EeXeGnyEqGmxGSan4nGRAFNLwTufLJmiYsTJ8j consumed 19293 of 200000 comput
e units
```

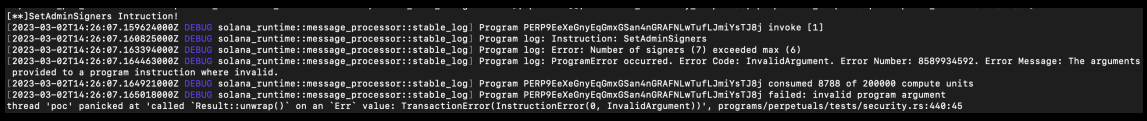
4.3 SETTING MORE ADMINS THAN ALLOWED

Description:

The Init and SetAdminsigners instructions set the administrator signers allowed to be part of the multisig. Both instructions need the new admin signers accounts to be provided as remaining accounts. It is also necessary to include a parameter, min_signatures, to set the minimum number of signatories required for an operation to be successfully carried out.
To achieve this, these instructions call the set_signers() function, providing the value of the min_signatures parameter and the remaining accounts mentioned. There is a maximum number of administrators that can be set, **Multisig::MAX_SIGNERS**. It has been tested to confirm no vulnerabilities were introduced, and the functionality is the expected.

Results:

No vulnerabilities were identified.



4.4 ADD COLLATERAL BY AN INCORRECT

Description:

It has been tested if the AddCollateral instruction's access control is correctly implemented or if otherwise, someone could add collateral to other user's position.

Results:

No vulnerabilities were identified.

```
[**]AddCollateral Instruction!
[2023-03-07T16:28:54.198977000Z DEBUG solana_runtime::message_processor::stable_log] Program ComputeBudget111111111111111111111111111 invoke [1]
[2023-03-07T16:28:54.199047000Z DEBUG solana_runtime::message_processor::stable_log] Program ComputeBudget111111111111111111111111111 success
[2023-03-07T16:28:54.214268000Z DEBUG solana_runtime::message_processor::stable_log] Program PERP9EeXeGnyEqGmxGSan4nGRAFNLWTufLJmiYsTJ8j invoke [1]
[2023-03-07T16:28:54.215646000Z DEBUG solana_runtime::message_processor::stable_log] Program log: Instruction: AddCollateral
[2023-03-07T16:28:54.225058000Z DEBUG solana_runtime::message_processor::stable_log] Program log: AnchorError caused by account: position. Error Code: ConstraintSeeds. Error Number: 2006. Error Message: A
 seeds constraint was violated.
[2023-03-07T16:28:54.225601000Z DEBUG solana_runtime::message_processor::stable_log] Program PERP9EeXeGnyEqGmxGSan4nGRAFNLWTufLJmiYsTJ8j consumed 20932 of 1400000 compute units
[2023-03-07T16:28:54.225705000Z DEBUG solana_runtime::message_processor::stable_log] Program PERP9EeXeGnyEqGmxGSan4nGRAFNLWTufLJmiYsTJ8j failed: custom program error: 0x7d6
thread 'poc' panicked at 'called `Result::unwrap()` on an `Err` value: TransactionError(InstructionError(1, Custom(2006)))', programs/perpetuals/tests/security.rs:2303:45
note: run with `RUST_BACKTRACE=1` environment variable to display a backtrace
test poc ... FAILED
```

4.5 REMOVE MORE LIQUIDITY THAN ADDED

Description:

The RemoveLiquidity instruction allows the user who added liquidity previously remove all or some of it from the custody token account. It has been checked if this functionality is safe or if otherwise, it is possible to remove more liquidity than added before and drain some funds.

Results:

No vulnerabilities were identified.

```
[2023-03-08T05:10:54.436347000Z DEBUG solana_runtime::message_processor::stable_log] Program TokenkegQfeZyiNwAJbNbGKPFXCWuBvf9Ss623VQ5DA invoke [2]
[2023-03-08T05:10:54.436995000Z DEBUG solana_runtime::message_processor::stable_log] Program log: Instruction: Transfer
[2023-03-08T05:10:54.439531000Z DEBUG solana_runtime::message_processor::stable_log] Program TokenkegQfeZyiNwAJbNbGKPFXCWuBvf9Ss623VQ5DA consumed 4786 of 1209741 compute units
[2023-03-08T05:10:54.439630000Z DEBUG solana_runtime::message_processor::stable_log] Program TokenkegQfeZyiNwAJbNbGKPFXCWuBvf9Ss623VQ5DA success
[2023-03-08T05:10:54.453381000Z DEBUG solana_runtime::message_processor::stable_log] Program TokenkegQfeZyiNwAJbNbGKPFXCWuBvf9Ss623VQ5DA invoke [2]
[2023-03-08T05:10:54.454025000Z DEBUG solana_runtime::message_processor::stable_log] Program log: Instruction: Burn
[2023-03-08T05:10:54.455897000Z DEBUG solana_runtime::message_processor::stable_log] Program log: Error: insufficient funds
[2023-03-08T05:10:54.456061000Z DEBUG solana_runtime::message_processor::stable_log] Program TokenkegQfeZyiNwAJbNbGKPFXCWuBvf9Ss623VQ5DA consumed 4198 of 1201797 compute units
[2023-03-08T05:10:54.456136000Z DEBUG solana_runtime::message_processor::stable_log] Program TokenkegQfeZyiNwAJbNbGKPFXCWuBvf9Ss623VQ5DA failed: custom program error: 0x1
[2023-03-08T05:10:54.456236000Z DEBUG solana_runtime::message_processor::stable_log] Program PERP9EeXeGnyEqGmxGSan4nGRAFNLwTufLJmiYsTJ8j consumed 202401 of 1400000 compute units
[2023-03-08T05:10:54.456309000Z DEBUG solana_runtime::message_processor::stable_log] Program PERP9EeXeGnyEqGmxGSan4nGRAFNLwTufLJmiYsTJ8j failed: custom program error: 0x1
```

4.6 CLOSE POSITION BY UNAUTHORIZED USER

Description:

The ClosePosition instruction allows the position's owner to close the position and transferring the tokens to a receiver. It has been tested that this instruction is safely implemented and nobody could close another user's position.

Results:

No vulnerabilities were identified.

```
[**]ClosePosition Intruction!
[2023-03-07T16:45:49.134858000Z DEBUG solana_runtime::message_processor::stable_log] Program ComputeBudget111111111111111111111111111 invoke [1]
[2023-03-07T16:45:49.134929000Z DEBUG solana_runtime::message_processor::stable_log] Program ComputeBudget111111111111111111111111111 success
[2023-03-07T16:45:49.150085000Z DEBUG solana_runtime::message_processor::stable_log] Program PERP9EeXeGnyEqGmxGSan4nGRAFNLwTufLJmiYsTJBj invoke [1]
[2023-03-07T16:45:49.151418000Z DEBUG solana_runtime::message_processor::stable_log] Program log: Instruction: ClosePosition
[2023-03-07T16:45:49.160944000Z DEBUG solana_runtime::message_processor::stable_log] Program log: AnchorError caused by account: position. Error Code: ConstraintSeeds. Error Number: 2006. Error Message: A
 seeds constraint was violated.
[2023-03-07T16:45:49.160994000Z DEBUG solana_runtime::message_processor::stable_log] Program log: Left:
[2023-03-07T16:45:49.161070000Z DEBUG solana_runtime::message_processor::stable_log] Program log: 7vHLuKxSZMwv1pgWyCdgrqcLrBEH8YT65Eq2mJTw5QJ8
[2023-03-07T16:45:49.161102000Z DEBUG solana_runtime::message_processor::stable_log] Program log: Right:
[2023-03-07T16:45:49.161170000Z DEBUG solana_runtime::message_processor::stable_log] Program log: 7yUAR3UCZGywbg81WZNced8Y6SwPr1QwzbJapkqcz9d8
[2023-03-07T16:45:49.161675000Z DEBUG solana_runtime::message_processor::stable_log] Program PERP9EeXeGnyEqGmxGSan4nGRAFNLwTufLJmiYsTJBj consumed 21636 of 1400000 compute units
[2023-03-07T16:45:49.161760000Z DEBUG solana_runtime::message_processor::stable_log] Program PERP9EeXeGnyEqGmxGSan4nGRAFNLwTufLJmiYsTJBj failed: custom program error: 0x7d6
thread 'poc' panicked at 'called `Result::unwrap()` on an `Err` value: TransactionError(InstructionError(1, Custom(2006)))', programs/perpetuals/tests/security.rs:2679:46
note: run with `RUST_BACKTRACE=1` environment variable to display a backtrace
```

AUTOMATED TESTING

5.1 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.

Results:

Soteria did not find any vulnerabilities.

```
Analyzing /workspace/perpetuals/programs/perpetuals/.coderrect/build/bpfel-unknown-unknown/release/all.ll ...
Cargo.toml: hasOverFlowChecks: 1ile
Cargo.toml: anchor_lang version: 0.26.0
anchor_lang_version: 0.26.0 anchorVersionTooOld: 0
 - ✔ [00m:03s] Loading IR From File
 - ▪ [00m:00s] Running Compiler Optimization Passes
EntryPoints:
entrypoint
 - ✔ [00m:00s] Running Compiler Optimization Passes
 - ✔ [00m:02s] Running Pointer Analysis
 - ✔ [00m:01s] Building Static Happens-Before Graph
 - ✔ [00m:00s] Detecting Vulnerabilities
detected 0 untrustful accounts in total.
detected 0 unsafe math operations in total.

--------The summary of potential vulnerabilities in all.ll--------

        No vulnerabilities detected
```

5.2 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.
RUSTSEC-2023-0001	tokio	reject_remote_clients Configuration corruption.

5.3 UNSAFE RUST CODE DETECTION

Description:

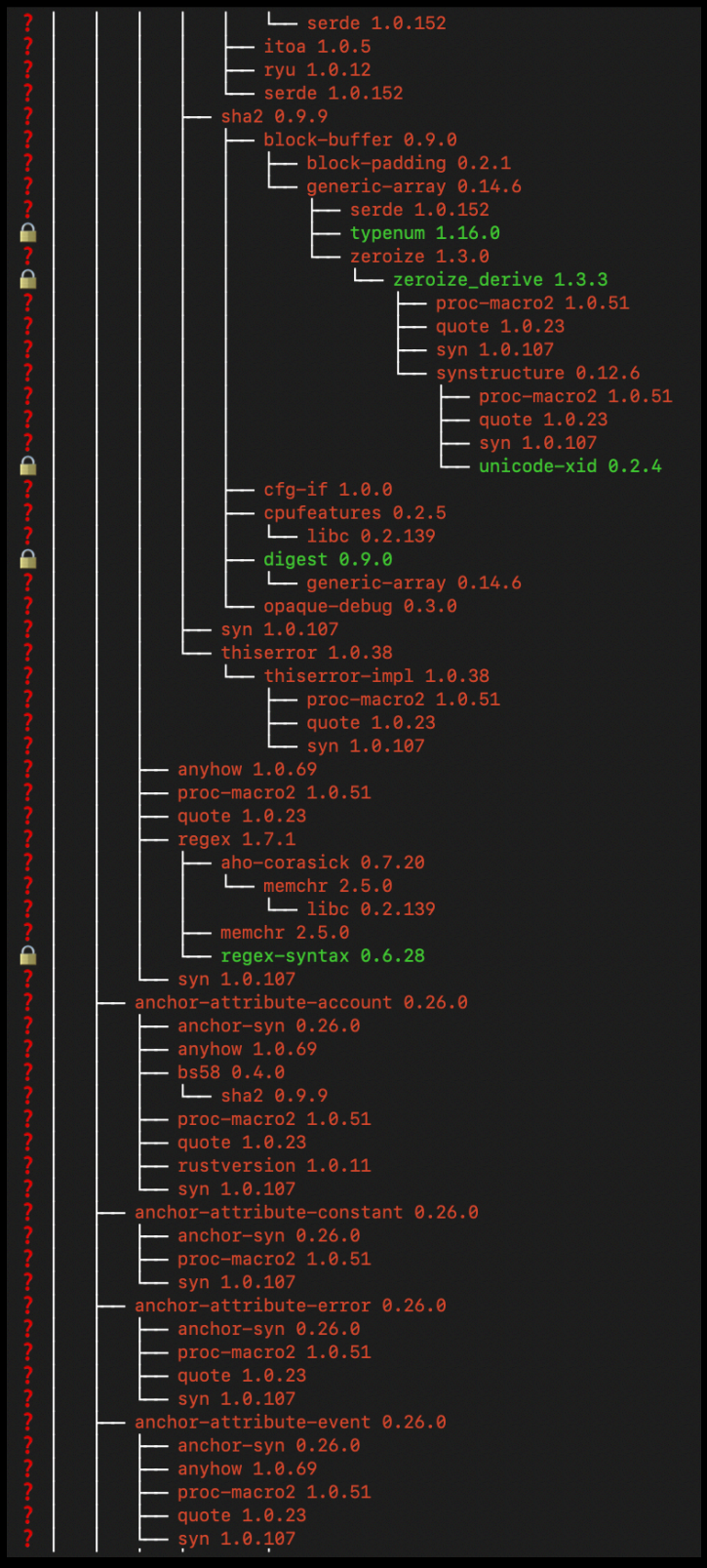
Halborn used automated security scanners to assist with the detection of well-known security issues and vulnerabilities. Among the tools used was cargo-geiger, a security tool that lists statistics related to the usage of unsafe Rust code in a core Rust codebase and all its dependencies.

Results:

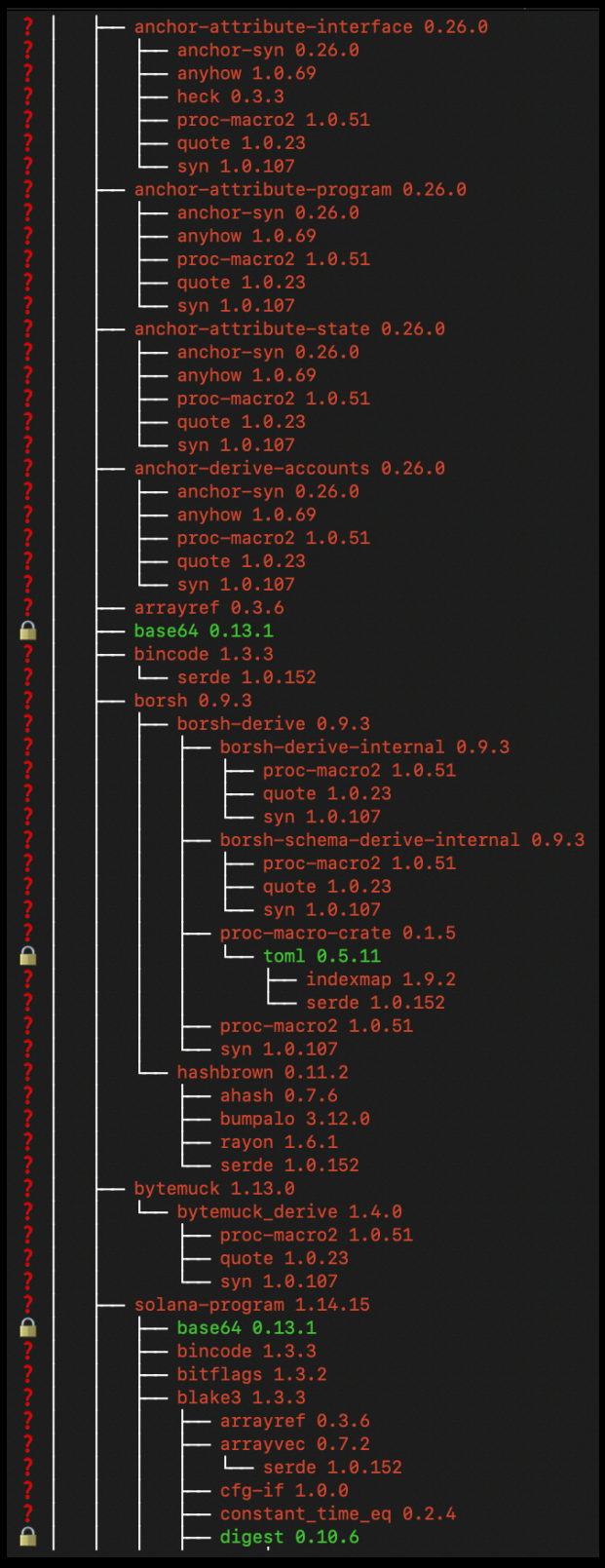
```
Symbols:
    🔒  = All entry point .rs files declare #![forbid(unsafe_code)].
    ❓  = This crate may use unsafe code.

❓ perpetuals 0.1.0
❓ ├── ahash 0.7.6
❓ │   ├── getrandom 0.2.8
❓ │   │   ├── cfg-if 1.0.0
❓ │   │   └── libc 0.2.139
❓ │   ├── once_cell 1.17.0
❓ │   │   └── parking_lot_core 0.9.7
❓ │   │       ├── cfg-if 1.0.0
❓ │   │       ├── libc 0.2.139
❓ │   │       └── smallvec 1.10.0
❓ │   │           └── serde 1.0.152
❓ │   │               └── serde_derive 1.0.152
❓ │   │                   ├── proc-macro2 1.0.51
❓ │   │                   │   └── unicode-ident 1.0.6
❓ │   │                   ├── quote 1.0.23
❓ │   │                   │   └── proc-macro2 1.0.51
❓ │   │                   └── syn 1.0.107
❓ │   │                       ├── proc-macro2 1.0.51
❓ │   │                       ├── quote 1.0.23
❓ │   │                       └── unicode-ident 1.0.6
❓ │   └── serde 1.0.152
❓ ├── anchor-lang 0.26.0
❓ │   ├── anchor-attribute-access-control 0.26.0
❓ │   │   ├── anchor-syn 0.26.0
❓ │   │   │   ├── anyhow 1.0.69
❓ │   │   │   ├── bs58 0.3.1
❓ │   │   │   ├── heck 0.3.3
❓ │   │   │   │   └── unicode-segmentation 1.10.1
❓ │   │   │   ├── proc-macro2 1.0.51
❓ │   │   │   ├── proc-macro2-diagnostics 0.9.1
❓ │   │   │   │   ├── proc-macro2 1.0.51
❓ │   │   │   │   ├── quote 1.0.23
❓ │   │   │   │   ├── syn 1.0.107
❓ │   │   │   │   └── yansi 0.5.1
❓ │   │   │   ├── quote 1.0.23
❓ │   │   │   ├── serde 1.0.152
❓ │   │   │   ├── serde_json 1.0.93
❓ │   │   │   │   ├── indexmap 1.9.2
❓ │   │   │   │   │   ├── hashbrown 0.12.3
❓ │   │   │   │   │   │   ├── ahash 0.7.6
❓ │   │   │   │   │   │   ├── bumpalo 3.12.0
❓ │   │   │   │   │   │   ├── rayon 1.6.1
❓ │   │   │   │   │   │   │   ├── either 1.8.1
❓ │   │   │   │   │   │   │   │   └── serde 1.0.152
❓ │   │   │   │   │   │   │   └── rayon-core 1.10.2
❓ │   │   │   │   │   │   │       ├── crossbeam-channel 0.5.6
❓ │   │   │   │   │   │   │       │   ├── cfg-if 1.0.0
❓ │   │   │   │   │   │   │       │   └── crossbeam-utils 0.8.14
❓ │   │   │   │   │   │   │       │       └── cfg-if 1.0.0
❓ │   │   │   │   │   │   │       ├── crossbeam-deque 0.8.2
❓ │   │   │   │   │   │   │       │   ├── cfg-if 1.0.0
❓ │   │   │   │   │   │   │       │   ├── crossbeam-epoch 0.9.13
❓ │   │   │   │   │   │   │       │   │   ├── cfg-if 1.0.0
❓ │   │   │   │   │   │   │       │   │   ├── crossbeam-utils 0.8.14
❓ │   │   │   │   │   │   │       │   │   ├── memoffset 0.7.1
❓ │   │   │   │   │   │   │       │   │   └── scopeguard 1.1.0
❓ │   │   │   │   │   │   │       │   └── crossbeam-utils 0.8.14
❓ │   │   │   │   │   │   │       ├── crossbeam-utils 0.8.14
❓ │   │   │   │   │   │   │       └── num_cpus 1.15.0
❓ │   │   │   │   │   │   │           └── libc 0.2.139
❓ │   │   │   │   │   │   └── serde 1.0.152
❓ │   │   │   │   │   └── rayon 1.6.1
```

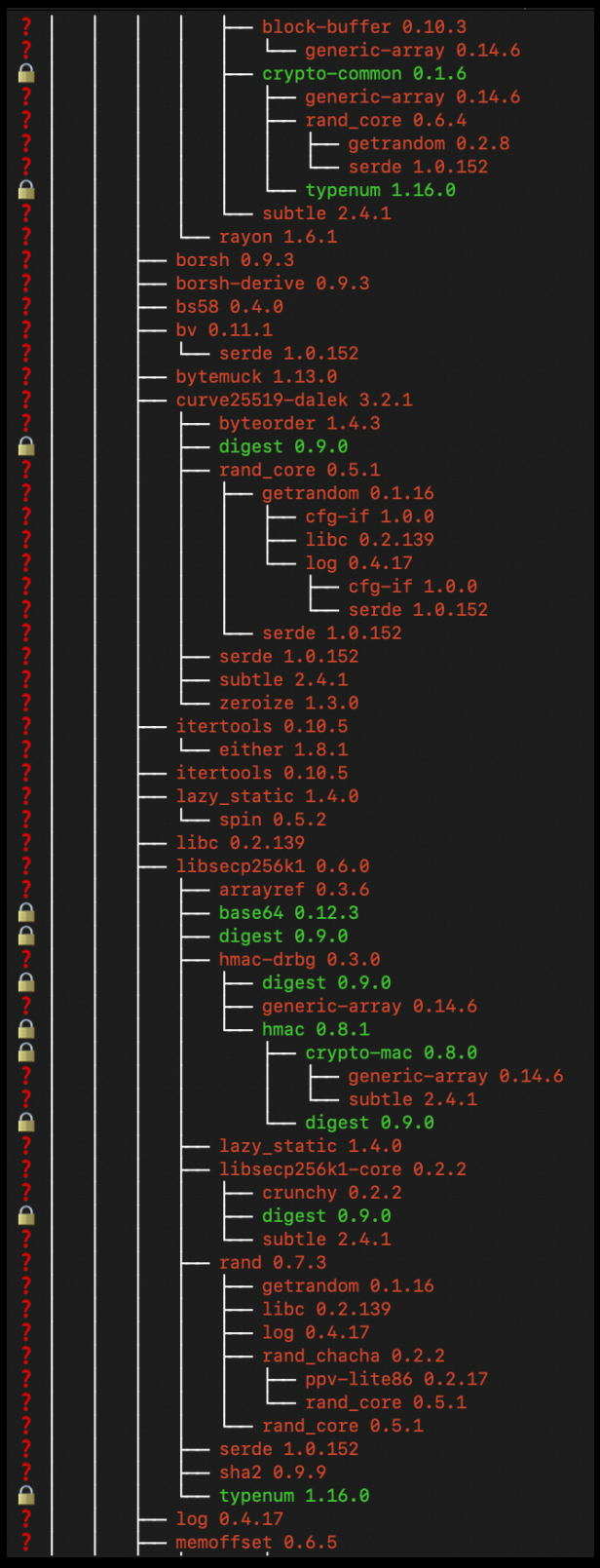
AUTOMATED TESTING



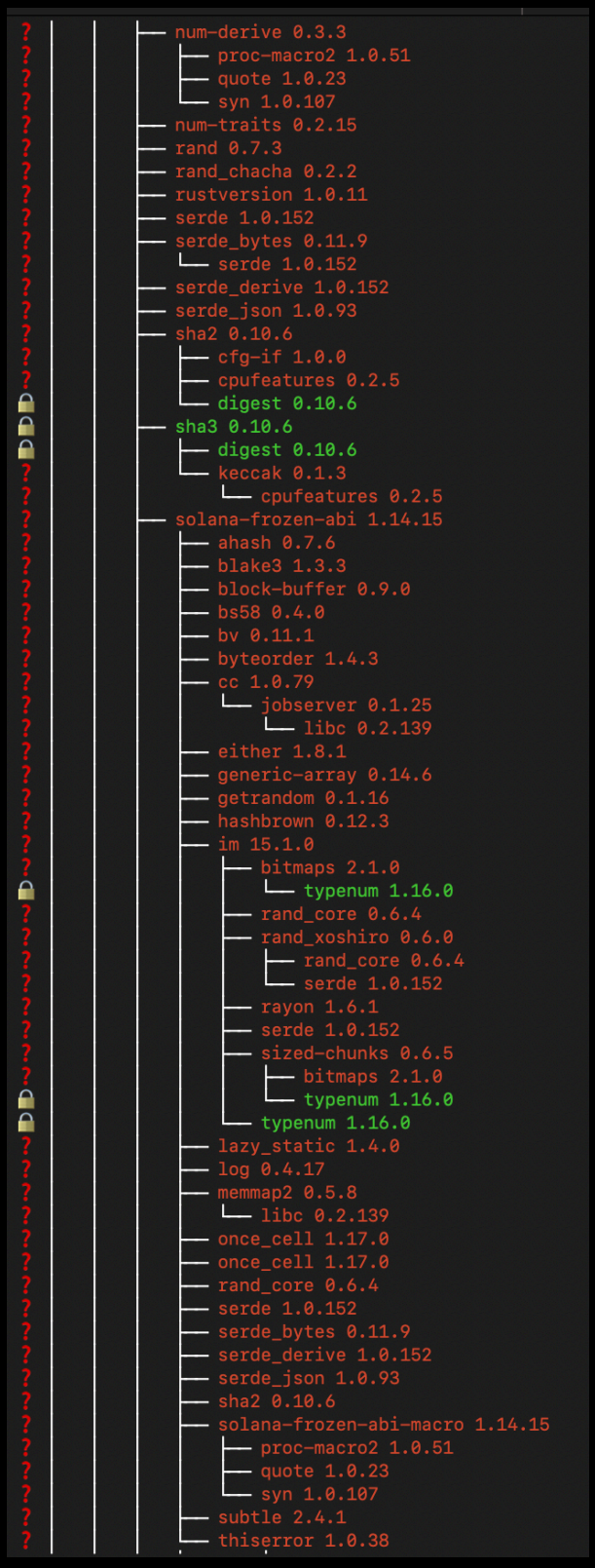
AUTOMATED TESTING



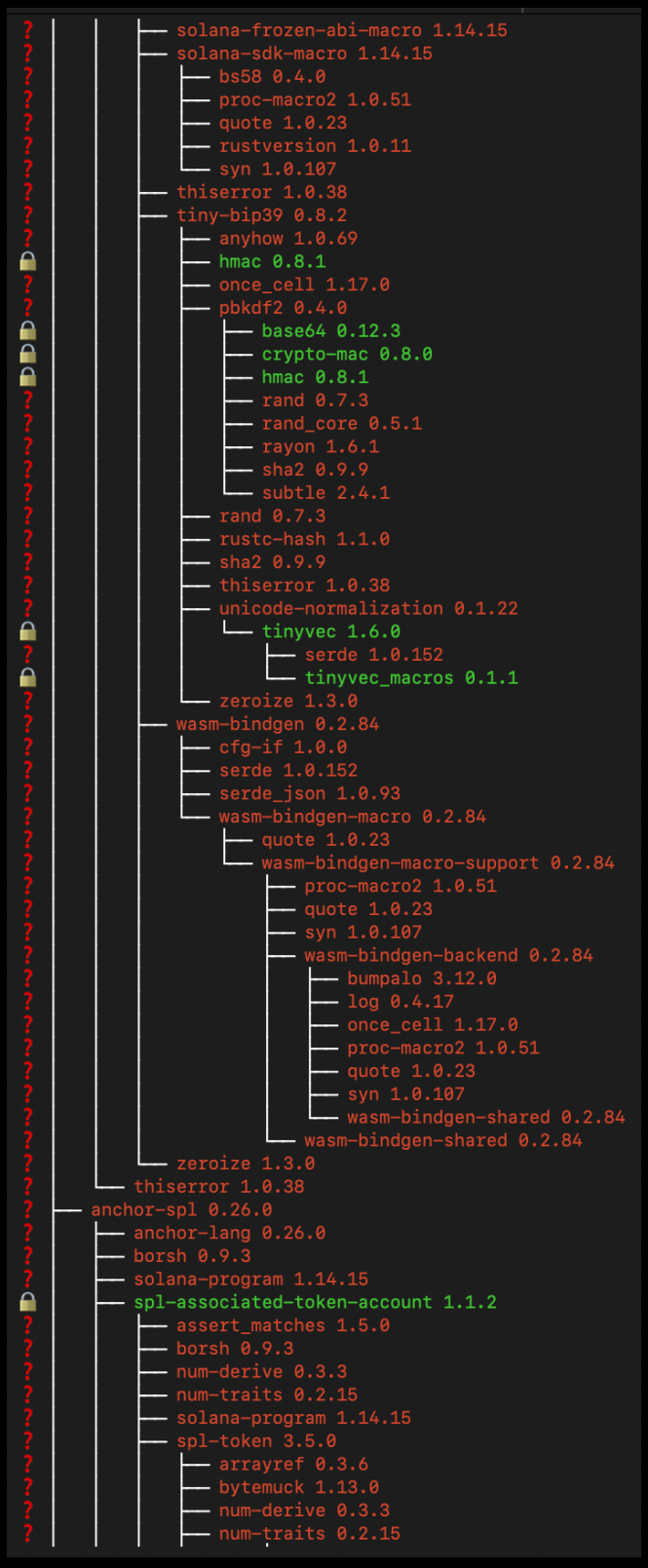
AUTOMATED TESTING



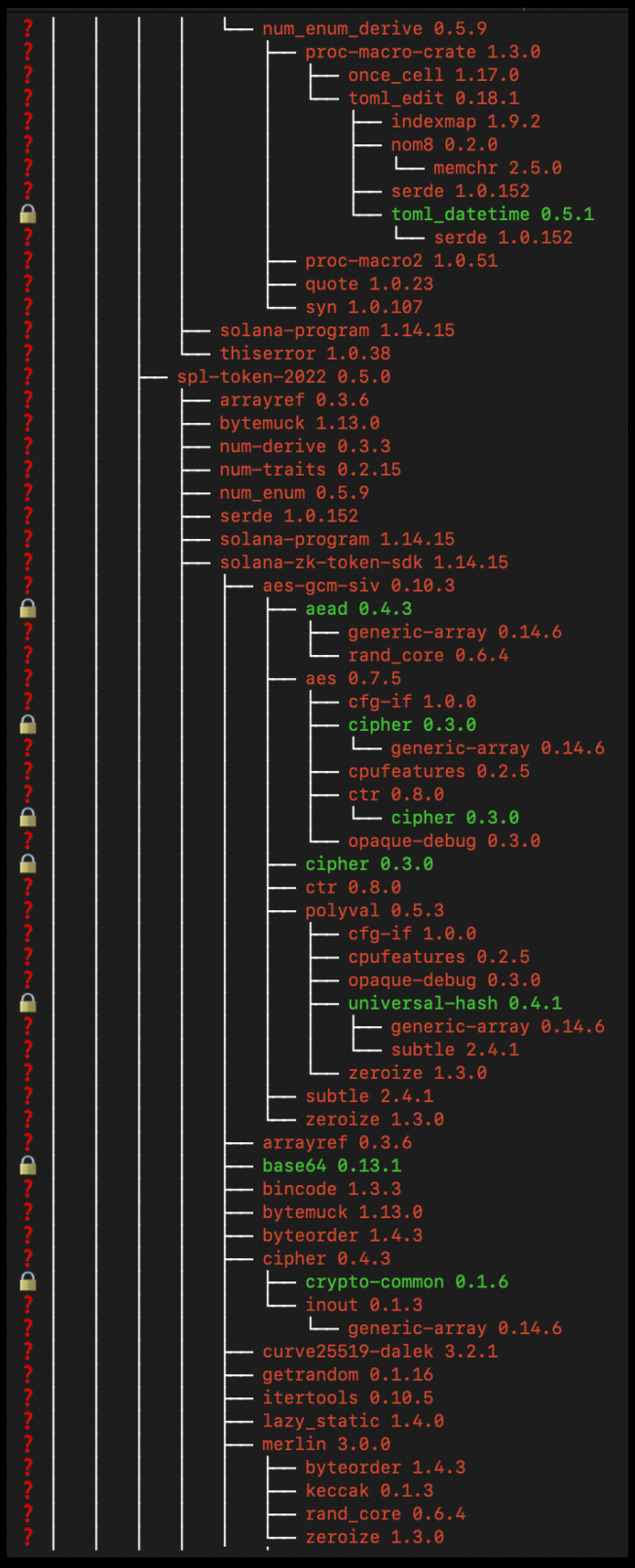
AUTOMATED TESTING



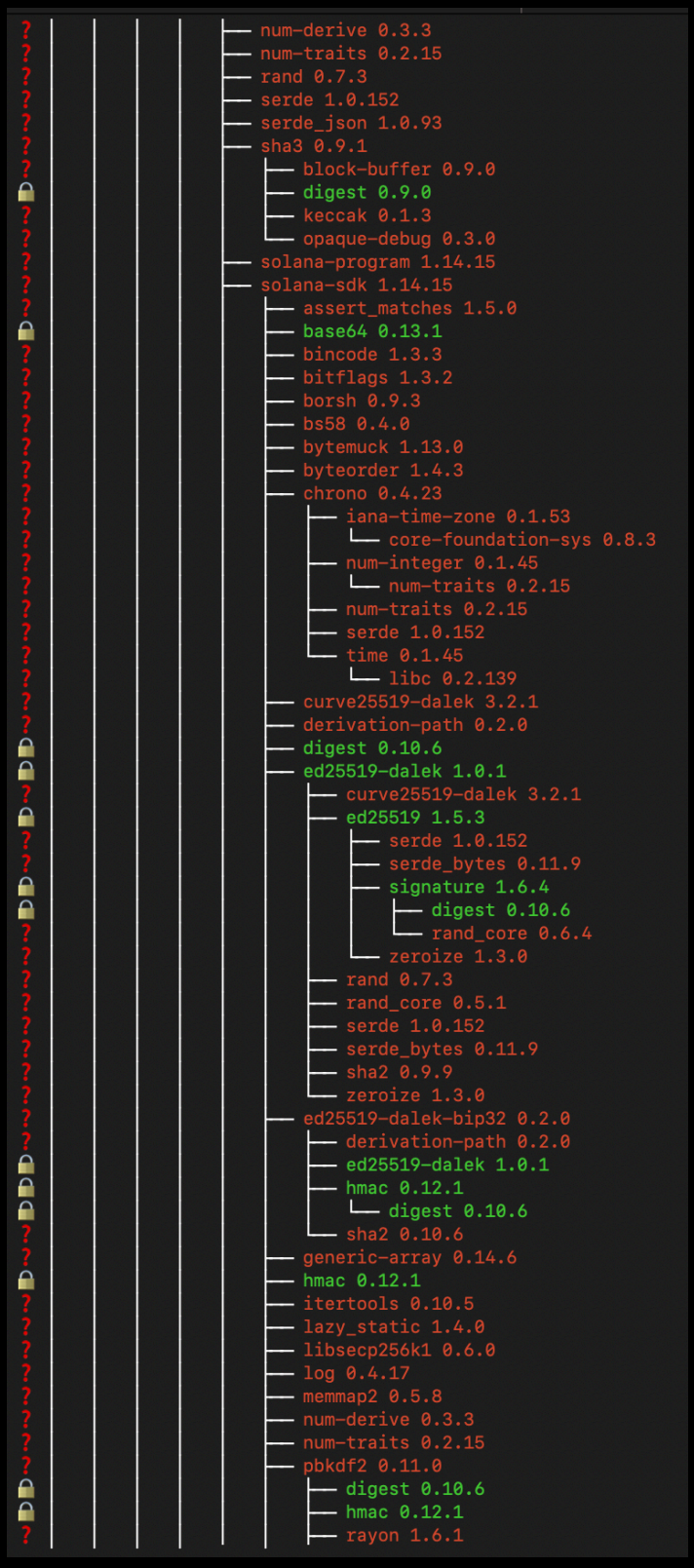
AUTOMATED TESTING



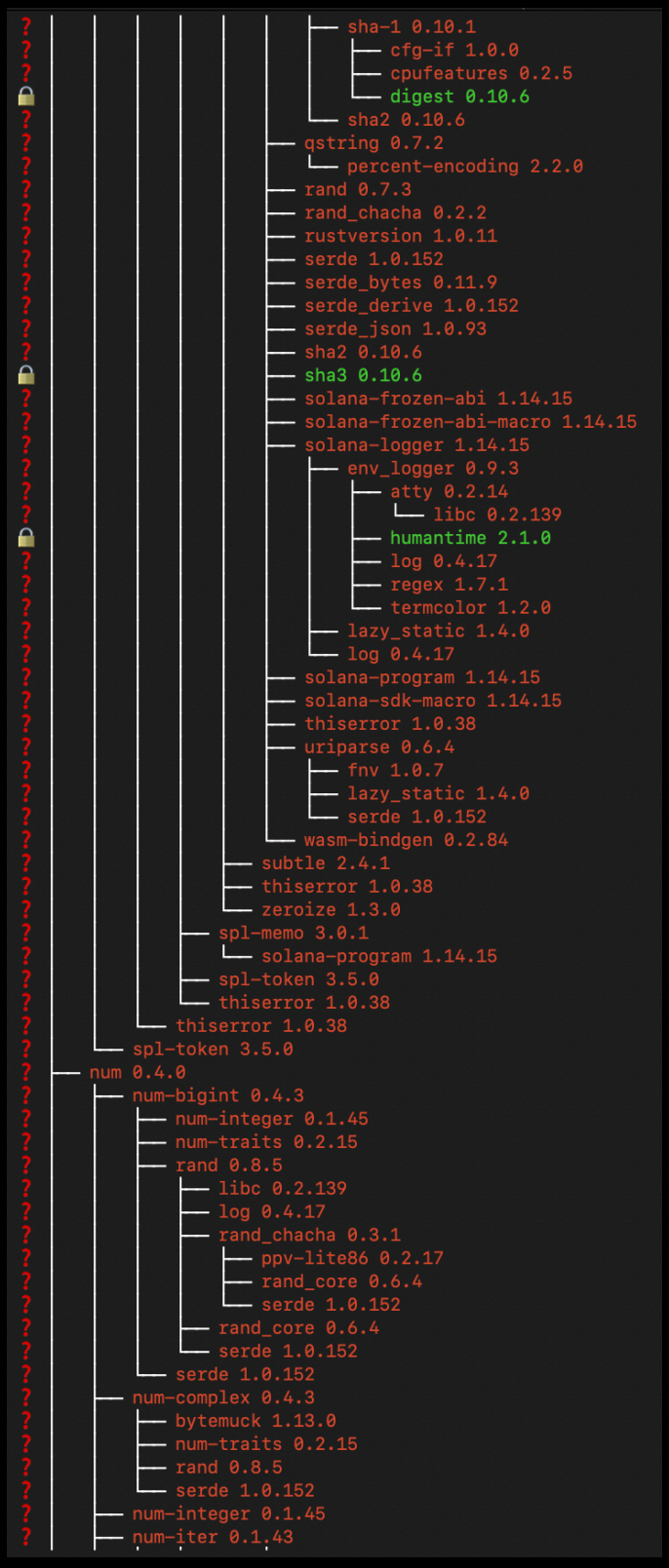
AUTOMATED TESTING



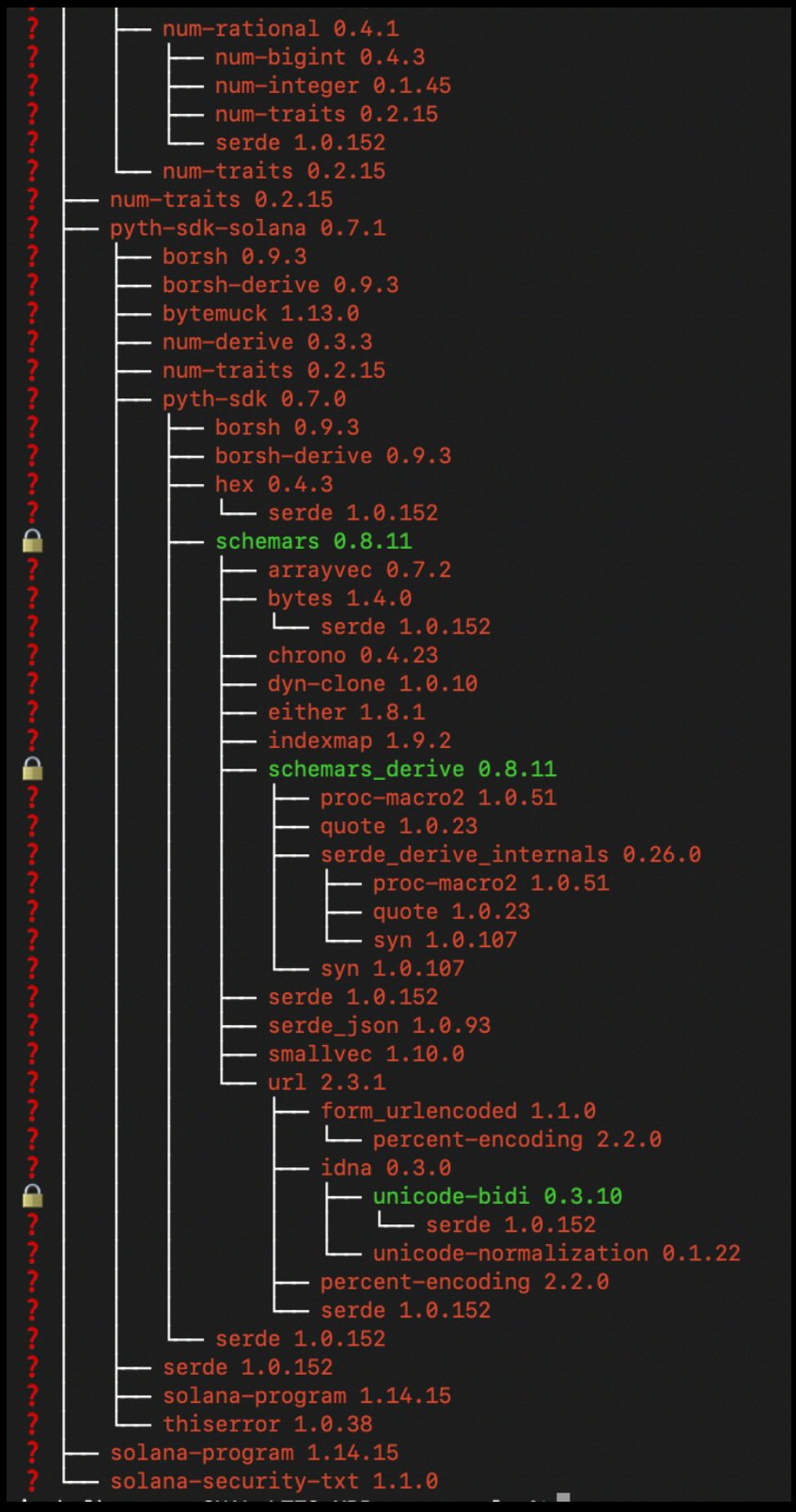
AUTOMATED TESTING



AUTOMATED TESTING



AUTOMATED TESTING



THANK YOU FOR CHOOSING

// HALBORN