



VRust

Security Assessment

O2Lab VRust Team

05/02/2022 21:18:58

Contents

Summary	5
Overview	6
Project Summary	6
Audit Summary	6
Vulnerability Summary	6
Findings	7
Issue: INT_CVE_0: IntegerCve - Overflow	10
Issue: INT_CVE_1: IntegerCve - Overflow	11
Issue: INT_CVE_2: IntegerCve - Overflow	12
Issue: CHK_CVE_0: MissingCheckerCve - is_signer	13
Issue: CHK_CVE_1: MissingCheckerCve - is_signer	14
Issue: CHK_CVE_2: MissingCheckerCve - is_signer	15
Issue: INT_CVE_3: IntegerCve - Overflow	16
Issue: CHK_CVE_3: MissingCheckerCve - is_signer	17
Issue: INT_CVE_4: IntegerCve - Overflow	18
Issue: CHK_CVE_4: MissingCheckerCve - is_signer	19
Issue: CHK_CVE_5: MissingCheckerCve - is_signer	20
Issue: CHK_CVE_6: MissingCheckerCve - is_signer	21
Issue: INT_CVE_5: IntegerCve - Overflow	22
Issue: INT_CVE_6: IntegerCve - Overflow	23
Issue: INT_CVE_7: IntegerCve - Overflow	24
Issue: INT_CVE_8: IntegerCve - Overflow	25
Issue: INT_CVE_9: IntegerCve - Overflow	26

Issue: INT_CVE_10: IntegerCve - Overflow	28
Issue: INT_CVE_11: IntegerCve - Overflow	30
Issue: INT_CVE_12: IntegerCve - Overflow	32
Issue: INT_CVE_13: IntegerCve - Overflow	34
Issue: CHK_CVE_7: MissingCheckerCve - is_signer	36
Issue: INT_CVE_14: IntegerCve - Overflow	37
Issue: INT_CVE_15: IntegerCve - Overflow	38
Issue: INT_CVE_16: IntegerCve - Overflow	39
Issue: INT_CVE_17: IntegerCve - Overflow	41
Issue: INT_CVE_18: IntegerCve - Overflow	43
Issue: INT_CVE_19: IntegerCve - Overflow	45
Issue: INT_CVE_20: IntegerCve - Overflow	47
Issue: CHK_CVE_8: MissingCheckerCve - is_signer	49
Issue: CHK_CVE_9: MissingCheckerCve - is_signer	50
Issue: CHK_CVE_10: MissingCheckerCve - is_signer	51
Issue: CHK_CVE_11: MissingCheckerCve - is_signer	52
Issue: INT_CVE_21: IntegerCve - Overflow	53
Issue: CHK_CVE_12: MissingCheckerCve - is_signer	54
Issue: CHK_CVE_13: MissingCheckerCve - is_signer	55
Issue: CHK_CVE_14: MissingCheckerCve - is_signer	56
Issue: CHK_CVE_15: MissingCheckerCve - is_signer	57
Issue: CHK_CVE_16: MissingCheckerCve - is_signer	58
Issue: CHK_CVE_17: MissingCheckerCve - is_signer	59

Appendix	60
Finding Categories	60
Gas Optimization	60
Mathematical Operations	60
Logical Issue	60
Language Specific	60
Coding Style	60
Checksum Calculation Method	60
Disclaimer	62

Summary

This report has been prepared for O2Lab VRust Team to discover issues and vulnerabilities in the source code of the O2Lab VRust Team project as well as any contract dependencies that were not part of an officially recognized library. A comprehensive examination has been performed, utilizing Static Analysis and Manual Review techniques. The auditing process pays special attention to the following considerations:

- Testing the smart contracts against both common and uncommon attack vectors.
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Ensuring contract logic meets the specifications and intentions of the client.
- Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- Thorough line-by-line manual review of the entire codebase by industry experts.

The security assessment resulted in findings that ranged from critical to informational. We recommend addressing these findings to ensure a high level of security standards and industry practices. We suggest recommendations that could better serve the project from the security perspective:

- Enhance general coding practices for better structures of source codes;
- Add enough unit tests to cover the possible use cases;
- Provide more comments per each function for readability, especially contracts that are verified in public;
- Provide more transparency on privileged activities once the protocol is live.

Overview

Project Summary

Project Name	O2Lab VRust Team
Platform	Ethereum
Language	Solana
Crate	clearing_house
GitHub Location	https://github.com/parasol-aser/vrust
sha256	Unknown

Audit Summary

Delivery Date	05/02/2022
Audit Methodology	Static Analysis
Key Components	

Vulnerability Summary

Vulnerability Level	Total
Critical	22
Major	0
Medium	0
Minor	0
Informational	18
Discussion	0

Findings

Bug Findings

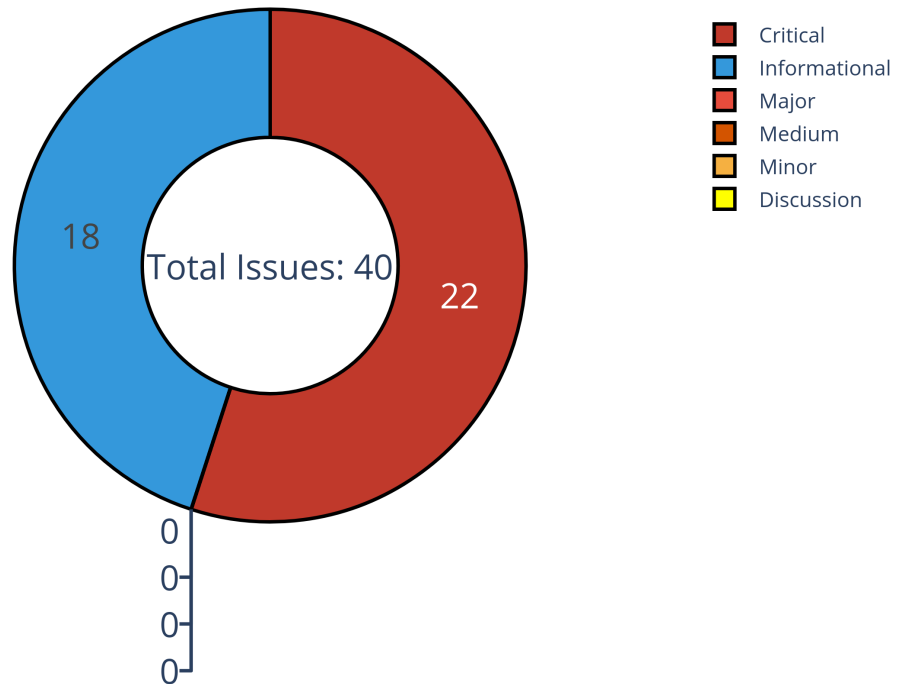


Figure 1: Findings

ID	Title	Category	Severity	Status
INT_CVE_0	Overflow	Integer Overflow	Critical	UnResolved
INT_CVE_1	Overflow	Integer Overflow	Critical	UnResolved
INT_CVE_2	Overflow	Integer Overflow	Critical	UnResolved
CHK_CVE_0	is_signer	Captured Signer Check	Informational	Resolved
CHK_CVE_1	is_signer	Captured Signer Check	Informational	Resolved
CHK_CVE_2	is_signer	Captured Signer Check	Informational	Resolved
INT_CVE_3	Overflow	Integer Overflow	Critical	UnResolved
CHK_CVE_3	is_signer	Captured Signer Check	Informational	Resolved

ID	Title	Category	Severity	Status
INT_CVE_4	Overflow	Integer Overflow	Critical	UnResolved
CHK_CVE_4	is_signer	Captured Signer Check	Informational	Resolved
CHK_CVE_5	is_signer	Captured Signer Check	Informational	Resolved
CHK_CVE_6	is_signer	Captured Signer Check	Informational	Resolved
INT_CVE_5	Overflow	Integer Overflow	Critical	UnResolved
INT_CVE_6	Overflow	Integer Overflow	Critical	UnResolved
INT_CVE_7	Overflow	Integer Overflow	Critical	UnResolved
INT_CVE_8	Overflow	Integer Overflow	Critical	UnResolved
INT_CVE_9	Overflow	Integer Overflow	Critical	UnResolved
INT_CVE_10	Overflow	Integer Overflow	Critical	UnResolved
INT_CVE_11	Overflow	Integer Overflow	Critical	UnResolved
INT_CVE_12	Overflow	Integer Overflow	Critical	UnResolved
INT_CVE_13	Overflow	Integer Overflow	Critical	UnResolved
CHK_CVE_7	is_signer	Captured Signer Check	Informational	Resolved
INT_CVE_14	Overflow	Integer Overflow	Critical	UnResolved
INT_CVE_15	Overflow	Integer Overflow	Critical	UnResolved
INT_CVE_16	Overflow	Integer Overflow	Critical	UnResolved
INT_CVE_17	Overflow	Integer Overflow	Critical	UnResolved
INT_CVE_18	Overflow	Integer Overflow	Critical	UnResolved
INT_CVE_19	Overflow	Integer Overflow	Critical	UnResolved
INT_CVE_20	Overflow	Integer Overflow	Critical	UnResolved
CHK_CVE_8	is_signer	Captured Signer Check	Informational	Resolved
CHK_CVE_9	is_signer	Captured Signer Check	Informational	Resolved
CHK_CVE_10	is_signer	Captured Signer Check	Informational	Resolved
CHK_CVE_11	is_signer	Captured Signer Check	Informational	Resolved
INT_CVE_21	Overflow	Integer Overflow	Critical	UnResolved
CHK_CVE_12	is_signer	Captured Signer Check	Informational	Resolved

ID	Title	Category	Severity	Status
CHK_CVE_13	is_signer	Captured Signer Check	Informational	Resolved
CHK_CVE_14	is_signer	Captured Signer Check	Informational	Resolved
CHK_CVE_15	is_signer	Captured Signer Check	Informational	Resolved
CHK_CVE_16	is_signer	Captured Signer Check	Informational	Resolved
CHK_CVE_17	is_signer	Captured Signer Check	Informational	Resolved

Issue: INT_CVE_0: IntegerCve - Overflow

Category	Severity	Status
Integer Overflow	Critical	UnResolved

- Location

programs/clearing_house/src/lib.rs:27:1: 27:11

```
27 #[program]
```

```
28
```

- Code Context

programs/clearing_house/src/lib.rs:27:1: 27:11

```
27 #[program]
```

```
28
```

- Call Stack

```
1 programs/clearing_house/src/lib.rs
```

- description:

Description of the bug here.

- link:

GitHub Link to be added.

- alleviation:

Some alleviation steps here.

Issue: INT_CVE_1: IntegerCve - Overflow

Category	Severity	Status
Integer Overflow	Critical	UnResolved

- Location

programs/clearing_house/src/lib.rs:27:1: 27:11

```
27  #[program]
```

```
28
```

- Code Context

programs/clearing_house/src/lib.rs:27:1: 27:11

```
27  #[program]
```

```
28
```

- Call Stack

```
1  programs/clearing_house/src/lib.rs
```

- description:

Description of the bug here.

- link:

GitHub Link to be added.

- alleviation:

Some alleviation steps here.

Issue: INT_CVE_2: IntegerCve - Overflow

Category	Severity	Status
Integer Overflow	Critical	UnResolved

- Location

programs/clearing_house/src/lib.rs:27:1: 27:11

```
27  #[program]
```

```
28
```

- Code Context

programs/clearing_house/src/lib.rs:27:1: 27:11

```
27  #[program]
```

```
28
```

- Call Stack

```
1  programs/clearing_house/src/lib.rs
```

- description:

Description of the bug here.

- link:

GitHub Link to be added.

- alleviation:

Some alleviation steps here.

Issue: CHK_CVE_0: MissingCheckerCve - is_signer

Category	Severity	Status
Captured Signer Check	Informational	Resolved

- Location

programs/clearing_house/src/instructions.rs:313:10: 313:18

313 Accounts
314

- Call Stack

```
1 <instructions::OpenPosition<'info> as  
  ↪ anchor_lang::Accounts<'info>>::try_accounts
```

- description:

Captured is_signer check for function: <instructions::OpenPosition<'info> as anchor_lang::Accounts<'info>>::try_accounts
We captured an is_signer check for variable: ::try_accounts

- link:

<https://github.com/parasol-aser/vrust/blob/yifei/patterns/01/README.md>

- alleviation:

Nothing needs to be done.

Issue: CHK_CVE_1: MissingCheckerCve - is_signer

Category	Severity	Status
Captured Signer Check	Informational	Resolved

- Location

programs/clearing_house/src/instructions.rs:78:10: 78:18

78 Accounts

79

- Call Stack

```
1 <instructions::InitializeUser<'info> as
  ↳ anchor_lang::Accounts<'info>>::try_accounts
```

- description:

Captured is_signer check for function: <instructions::InitializeUser<'info> as anchor_lang::Accounts<'info>>::try_accounts
We captured an is_signer check for variable: ::try_accounts

- link:

<https://github.com/parasol-aser/vrust/blob/yifei/patterns/01/README.md>

- alleviation:

Nothing needs to be done.

Issue: CHK_CVE_2: MissingCheckerCve - is_signer

Category	Severity	Status
Captured Signer Check	Informational	Resolved

- Location

programs/clearing_house/src/instructions.rs:230:10: 230:18

230
231

Accounts

- Call Stack

```
1 <instructions::WithdrawFees<'info> as  
  ↳ anchor_lang::Accounts<'info>::try_accounts
```

- description:

Captured is_signer check for function: <instructions::WithdrawFees<'info> as anchor_lang::Accounts<'info>::try_accounts>
We captured an is_signer check for variable: ::try_accounts

- link:

<https://github.com/parasol-aser/vrust/blob/yifei/patterns/01/README.md>

- alleviation:

Nothing needs to be done.

Issue: INT_CVE_3: IntegerCve - Overflow

Category	Severity	Status
Integer Overflow	Critical	UnResolved

- Location

/home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-0.9.1/src/uint.rs:834:31: 834:63

```
834 (Self::WORD_BITS as u32 - shift)
835
```

- Code Context

/home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-0.9.1/src/uint.rs:825:4: 838:5

```
825 fn full_shr(u: [u64; $n_words + 1], shift: u32) -> Self {
    ↳ debug_assert!(shift < Self::WORD_BITS as u32);           let mut res =
    ↳ Self::zero();           for i in 0..$n_words {           res.0[i] =
    ↳ u[i] >> shift;           }           // carry           if shift > 0 {
    ↳ for i in 1..=$n_words {           res.0[i - 1] |= u[i] <<
    ↳ (Self::WORD_BITS as u32 - shift);           }           }
    ↳ res           }
```

- Call Stack

```
1 /home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-
  ↳ 0.9.1/src/uint.rs
```

- description:

Description of the bug here.

- link:

GitHub Link to be added.

- alleviation:

Some alleviation steps here.

Issue: CHK_CVE_3: MissingCheckerCve - is_signer

Category	Severity	Status
Captured Signer Check	Informational	Resolved

- Location

programs/clearing_house/src/instructions.rs:137:10: 137:18

```
137 Accounts
138
```

- Call Stack

```
1 <instructions::DepositCollateral<'info> as
  ↳ anchor_lang::Accounts<'info>>::try_accounts
```

- description:

Captured is_signer check for function: <instructions::DepositCollateral<'info> as anchor_lang::Accounts<'info>>::try_accounts
We captured an is_signer check for variable: ::try_accounts

- link:

<https://github.com/parasol-aser/vrust/blob/yifei/patterns/01/README.md>

- alleviation:

Nothing needs to be done.

Issue: INT_CVE_4: IntegerCve - Overflow

Category	Severity	Status
Integer Overflow	Critical	UnResolved

- Location

/home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-0.9.1/src/uint.rs:819:24: 819:56

```
819 (Self::WORD_BITS as u32 - shift)
820
```

- Code Context

/home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-0.9.1/src/uint.rs:815:4: 823:5

```
815 fn full_shl(self, shift: u32) -> [u64; $n_words + 1] {
    ↪ debug_assert!(shift < Self::WORD_BITS as u32);          let mut u =
    ↪ [0u64; $n_words + 1];          let u_lo = self.0[0] << shift;
    ↪ let u_hi = self >> (Self::WORD_BITS as u32 - shift);      u[0] =
    ↪ u_lo;          u[1..].copy_from_slice(&u_hi.0[..]);      u
    ↪ }
816
```

- Call Stack

```
1 /home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-
  ↪ 0.9.1/src/uint.rs
```

- description:

Description of the bug here.

- link:

GitHub Link to be added.

- alleviation:

Some alleviation steps here.

Issue: CHK_CVE_4: MissingCheckerCve - is_signer

Category	Severity	Status
Captured Signer Check	Informational	Resolved

- Location

programs/clearing_house/src/instructions.rs:391:10: 391:18

```
391 Accounts
392
```

- Call Stack

```
1 <instructions::Liquidate<'info> as
  ↳ anchor_lang::Accounts<'info>>::try_accounts
```

- description:

Captured is_signer check for function: <instructions::Liquidate<'info> as anchor_lang::Accounts<'info>>::try_accounts
We captured an is_signer check for variable: ::try_accounts

- link:

<https://github.com/parasol-aser/vrust/blob/yifei/patterns/01/README.md>

- alleviation:

Nothing needs to be done.

Issue: CHK_CVE_5: MissingCheckerCve - is_signer

Category	Severity	Status
Captured Signer Check	Informational	Resolved

- Location

programs/clearing_house/src/instructions.rs:13:10: 13:18

13 Accounts

14

- Call Stack

```
1 <instructions::Initialize<'info> as
  ↳ anchor_lang::Accounts<'info>>::try_accounts
```

- description:

Captured is_signer check for function: <instructions::Initialize<'info> as anchor_lang::Accounts<'info>>::try_accounts
We captured an is_signer check for variable: ::try_accounts

- link:

<https://github.com/parasol-aser/vrust/blob/yifei/patterns/01/README.md>

- alleviation:

Nothing needs to be done.

Issue: CHK_CVE_6: MissingCheckerCve - is_signer

Category	Severity	Status
Captured Signer Check	Informational	Resolved

- Location

programs/clearing_house/src/instructions.rs:536:10: 536:18

```
536 Accounts
537
```

- Call Stack

```
1 <instructions::AdminUpdateK<'info> as
  ↳ anchor_lang::Accounts<'info>>::try_accounts
```

- description:

Captured is_signer check for function: <instructions::AdminUpdateK<'info> as anchor_lang::Accounts<'info>>::try_accounts
We captured an is_signer check for variable: ::try_accounts

- link:

<https://github.com/parasol-aser/vrust/blob/yifei/patterns/01/README.md>

- alleviation:

Nothing needs to be done.

Issue: INT_CVE_5: IntegerCve - Overflow

Category	Severity	Status
Integer Overflow	Critical	UnResolved

- Location

/home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-0.9.1/src/uint.rs:834:31: 834:63

```
834 (Self::WORD_BITS as u32 - shift)
835
```

- Code Context

/home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-0.9.1/src/uint.rs:825:4: 838:5

```
825 fn full_shr(u: [u64; $n_words + 1], shift: u32) -> Self {
  ↳ debug_assert!(shift < Self::WORD_BITS as u32);           let mut res =
  ↳ Self::zero();           for i in 0..$n_words {           res.0[i] =
  ↳ u[i] >> shift;           }           // carry           if shift > 0 {
  ↳ for i in 1..=$n_words {           res.0[i - 1] |= u[i] <<
  ↳ (Self::WORD_BITS as u32 - shift);           }           }
  ↳ res           }
```

826

- Call Stack

```
1 /home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-
  ↳ 0.9.1/src/uint.rs
```

- description:

Description of the bug here.

- link:

GitHub Link to be added.

- alleviation:

Some alleviation steps here.

Issue: INT_CVE_6: IntegerCve - Overflow

Category	Severity	Status
Integer Overflow	Critical	UnResolved

- Location

/home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-0.9.1/src/uint.rs:819:24: 819:56

```
819 (Self::WORD_BITS as u32 - shift)
820
```

- Code Context

/home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-0.9.1/src/uint.rs:815:4: 823:5

```
815 fn full_shl(self, shift: u32) -> [u64; $n_words + 1] {
    ↪ debug_assert!(shift < Self::WORD_BITS as u32);          let mut u =
    ↪ [0u64; $n_words + 1];          let u_lo = self.0[0] << shift;
    ↪ let u_hi = self >> (Self::WORD_BITS as u32 - shift);      u[0] =
    ↪ u_lo;          u[1..].copy_from_slice(&u_hi.0[..]);      u
    ↪ }
816
```

- Call Stack

```
1 /home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-
  ↪ 0.9.1/src/uint.rs
```

- description:

Description of the bug here.

- link:

GitHub Link to be added.

- alleviation:

Some alleviation steps here.

Issue: INT_CVE_7: IntegerCve - Overflow

Category	Severity	Status
Integer Overflow	Critical	UnResolved

- Location

/home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-0.9.1/src/uint.rs:1250:37:1250:58

```
1250 a as u128 * b as u128
1251
```

- Code Context

/home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-0.9.1/src/uint.rs:1249:4: 1252:5

```
1249 const fn mul_u64(a: u64, b: u64, carry: u64) -> (u64, u64) { let
↳ (hi, lo) = Self::split_u128(a as u128 * b as u128 + carry as u128);
↳ (lo, hi) }
1250
```

- Call Stack

```
1 /home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-
↳ 0.9.1/src/uint.rs
```

- description:

Description of the bug here.

- link:

GitHub Link to be added.

- alleviation:

Some alleviation steps here.

Issue: INT_CVE_8: IntegerCve - Overflow

Category	Severity	Status
Integer Overflow	Critical	UnResolved

- Location

/home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-0.9.1/src/uint.rs:1250:37:1250:74

```
1250 a as u128 * b as u128 + carry as u128
1251
```

- Code Context

/home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-0.9.1/src/uint.rs:1249:4: 1252:5

```
1249 const fn mul_u64(a: u64, b: u64, carry: u64) -> (u64, u64) { let
↳ (hi, lo) = Self::split_u128(a as u128 * b as u128 + carry as u128);
↳ (lo, hi) }
1250
```

- Call Stack

```
1 /home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-
↳ 0.9.1/src/uint.rs
```

- description:

Description of the bug here.

- link:

GitHub Link to be added.

- alleviation:

Some alleviation steps here.

Issue: INT_CVE_9: IntegerCve - Overflow

Category	Severity	Status
Integer Overflow	Critical	UnResolved

• Location

/home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-0.9.1/src/uint.rs:879:19: 879:24

```
879 j + n
880
```

• Code Context

/home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-0.9.1/src/uint.rs:859:4: 939:5

```
859 fn div_mod_knuth(self, mut v: Self, n: usize, m: usize) -> (Self, Self) {
    ↪ debug_assert!(self.bits() >= v.bits() && !v.fits_word());
    ↪ debug_assert!(n + m <= $n_words);           // D1.           // Make
    ↪ sure 64th bit in v's highest word is set.    // If we shift both
    ↪ self and v, it won't affect the quotient     // and the
    ↪ remainder will only need to be shifted back. let shift = v.0[n
    ↪ - 1].leading_zeros(); v <=< shift;           // u will store
    ↪ the remainder (shifted) let mut u = self.full_shl(shift);
860     // quotient let mut q = Self::zero(); let
    ↪ v_n_1 = v.0[n - 1]; let v_n_2 = v.0[n - 2];
861     // D2. D7. // iterate from m downto 0 for j
    ↪ in (0..=m).rev() { let u_jn = u[j + n];
862     // D3. // q_hat is our guess for the j-th
    ↪ quotient digit // q_hat = min(b - 1,
    ↪ (u_{j+n} * b + u_{j+n-1}) / v_{n-1}) // b
    ↪ = 1 << WORD_BITS // Theorem B: q_hat >=
    ↪ q_j >= q_hat - 2 let mut q_hat = if u_jn <
    ↪ v_n_1 { let (mut q_hat, mut r_hat) =
    ↪ Self::div_mod_word(u_jn, u[j + n - 1], v_n_1);
    ↪ // this loop takes at most 2 iterations
    ↪ loop { // check if q_hat * v_{n-2}
    ↪ > b * r_hat + u_{j+n-2} let (hi,
    ↪ lo) = Self::split_u128(u128::from(q_hat) *
    ↪ u128::from(v_n_2)); if (hi, lo) <=
    ↪ (r_hat, u[j + n - 2]) { break;
    ↪ } // then iterate till it doesn't
    ↪ hold q_hat -= 1;
    ↪ let (new_r_hat, overflow) =
O2Lab VRust Team r_hat.overflowing_add(v_n_1); 26
    ↪ r_hat = new_r_hat; // if r_hat
    ↪ overflowed, we're done if overflow
    ↪ { break;
    ↪ } q_hat
    ↪ } // then iterate till it doesn't
```

```

863      // ex. 20:          // since q_hat * v_{n-2} <= b *
      ↪ r_hat + u_{j+n-2},          // either q_hat == q_j,
      ↪ or q_hat == q_j + 1
864      // D4.             // let's assume optimistically q_hat ==
      ↪ q_j                 // subtract (q_hat * v) from u[j..]
      ↪ let q_hat_v = v.full_mul_u64(q_hat);          //
      ↪ u[j..] -= q_hat_v;          let c =
      ↪ Self::sub_slice(&mut u[j..], &q_hat_v[..n + 1]);
865      // D6.             // actually, q_hat == q_j + 1 and
      ↪ u[j..] has overflowed          // highly unlikely ~
      ↪ (1 / 2^63)          if c {          q_hat -=
      ↪ 1;                  // add v to u[j..]
      ↪ let c = Self::add_slice(&mut u[j..], &v.0[..n]);
      ↪ u[j + n] = u[j + n].wrapping_add(u64::from(c));
      ↪ }
866      // D5.             q.0[j] = q_hat;          }
867      // D8.             let remainder = Self::full_shr(u, shift);
868      (q, remainder)      }
869

```

- Call Stack

```

1  /home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-
  ↪ 0.9.1/src/uint.rs

```

- description:

Description of the bug here.

- link:

GitHub Link to be added.

- alleviation:

Some alleviation steps here.

Issue: INT_CVE_10: IntegerCve - Overflow

Category	Severity	Status
Integer Overflow	Critical	UnResolved

• Location

/home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-0.9.1/src/uint.rs:887:63: 887:68

```
887 j + n
888
```

• Code Context

/home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-0.9.1/src/uint.rs:859:4: 939:5

```
859 fn div_mod_knuth(self, mut v: Self, n: usize, m: usize) -> (Self, Self) {
    ↪ debug_assert!(self.bits() >= v.bits() && !v.fits_word());
    ↪ debug_assert!(n + m <= $n_words);           // D1.           // Make
    ↪ sure 64th bit in v's highest word is set.    // If we shift both
    ↪ self and v, it won't affect the quotient     // and the
    ↪ remainder will only need to be shifted back. let shift = v.0[n
    ↪ - 1].leading_zeros(); v <=< shift;           // u will store
    ↪ the remainder (shifted) let mut u = self.full_shl(shift);
860 // quotient let mut q = Self::zero(); let
    ↪ v_n_1 = v.0[n - 1]; let v_n_2 = v.0[n - 2];
861 // D2. D7. // iterate from m downto 0 for j
    ↪ in (0..=m).rev() { let u_jn = u[j + n];
862 // D3. // q_hat is our guess for the j-th
    ↪ quotient digit // q_hat = min(b - 1,
    ↪ (u_{j+n} * b + u_{j+n-1}) / v_{n-1}) // b
    ↪ = 1 << WORD_BITS // Theorem B: q_hat >=
    ↪ q_j >= q_hat - 2 let mut q_hat = if u_jn <
    ↪ v_n_1 { let (mut q_hat, mut r_hat) =
    ↪ Self::div_mod_word(u_jn, u[j + n - 1], v_n_1);
    ↪ // this loop takes at most 2 iterations
    ↪ loop { // check if q_hat * v_{n-2}
    ↪ > b * r_hat + u_{j+n-2} let (hi,
    ↪ lo) = Self::split_u128(u128::from(q_hat) *
    ↪ u128::from(v_n_2)); if (hi, lo) <=
    ↪ (r_hat, u[j + n - 2]) { break;
    ↪ } // then iterate till it doesn't
    ↪ hold q_hat -= 1;
    ↪ let (new_r_hat, overflow) =
O2Lab VRust Team r_hat.overflowing_add(v_n_1); 28
    ↪ r_hat = new_r_hat; // if r_hat
    ↪ overflowed, we're done if overflow
    ↪ { break;
    ↪ } q_hat
    ↪ } // (q_hat, r_hat) = (q_hat, r_hat)
    ↪ } // (q_hat, r_hat) = (q_hat, r_hat)
```

```

863      // ex. 20:          // since q_hat * v_{n-2} <= b *
      ↪ r_hat + u_{j+n-2},          // either q_hat == q_j,
      ↪ or q_hat == q_j + 1
864      // D4.             // let's assume optimistically q_hat ==
      ↪ q_j                 // subtract (q_hat * v) from u[j..]
      ↪ let q_hat_v = v.full_mul_u64(q_hat);          //
      ↪ u[j..] -= q_hat_v;          let c =
      ↪ Self::sub_slice(&mut u[j..], &q_hat_v[..n + 1]);
865      // D6.             // actually, q_hat == q_j + 1 and
      ↪ u[j..] has overflowed          // highly unlikely ~
      ↪ (1 / 2^63)          if c {          q_hat -=
      ↪ 1;                  // add v to u[j..]
      ↪ let c = Self::add_slice(&mut u[j..], &v.0[..n]);
      ↪ u[j + n] = u[j + n].wrapping_add(u64::from(c));
      ↪ }
866      // D5.             q.0[j] = q_hat;          }
867      // D8.             let remainder = Self::full_shr(u, shift);
868      (q, remainder)      }
869

```

- Call Stack

```

1  /home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-
  ↪ 0.9.1/src/uint.rs

```

- description:

Description of the bug here.

- link:

GitHub Link to be added.

- alleviation:

Some alleviation steps here.

Issue: INT_CVE_11: IntegerCve - Overflow

Category	Severity	Status
Integer Overflow	Critical	UnResolved

• Location

/home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-0.9.1/src/uint.rs:892:33: 892:38

```
892 j + n
893
```

• Code Context

/home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-0.9.1/src/uint.rs:859:4: 939:5

```
859 fn div_mod_knuth(self, mut v: Self, n: usize, m: usize) -> (Self, Self) {
    ↪ debug_assert!(self.bits() >= v.bits() && !v.fits_word());
    ↪ debug_assert!(n + m <= $n_words);           // D1.           // Make
    ↪ sure 64th bit in v's highest word is set.    // If we shift both
    ↪ self and v, it won't affect the quotient     // and the
    ↪ remainder will only need to be shifted back. let shift = v.0[n
    ↪ - 1].leading_zeros(); v <=< shift;           // u will store
    ↪ the remainder (shifted) let mut u = self.full_shl(shift);
860 // quotient let mut q = Self::zero(); let
    ↪ v_n_1 = v.0[n - 1]; let v_n_2 = v.0[n - 2];
861 // D2. D7. // iterate from m downto 0 for j
    ↪ in (0..=m).rev() { let u_jn = u[j + n];
862 // D3. // q_hat is our guess for the j-th
    ↪ quotient digit // q_hat = min(b - 1,
    ↪ (u_{j+n} * b + u_{j+n-1}) / v_{n-1}) // b
    ↪ = 1 << WORD_BITS // Theorem B: q_hat >=
    ↪ q_j >= q_hat - 2 let mut q_hat = if u_jn <
    ↪ v_n_1 { let (mut q_hat, mut r_hat) =
    ↪ Self::div_mod_word(u_jn, u[j + n - 1], v_n_1);
    ↪ // this loop takes at most 2 iterations
    ↪ loop { // check if q_hat * v_{n-2}
    ↪ > b * r_hat + u_{j+n-2} let (hi,
    ↪ lo) = Self::split_u128(u128::from(q_hat) *
    ↪ u128::from(v_n_2)); if (hi, lo) <=
    ↪ (r_hat, u[j + n - 2]) { break;
    ↪ } // then iterate till it doesn't
    ↪ hold q_hat -= 1;
    ↪ let (new_r_hat, overflow) =
O2Lab VRust Team r_hat.overflowing_add(v_n_1); 30
    ↪ r_hat = new_r_hat; // if r_hat
    ↪ overflowed, we're done if overflow
    ↪ { break;
    ↪ } q_hat
    ↪ } }
```

```

863      // ex. 20:          // since q_hat * v_{n-2} <= b *
      ↪ r_hat + u_{j+n-2},          // either q_hat == q_j,
      ↪ or q_hat == q_j + 1
864      // D4.             // let's assume optimistically q_hat ==
      ↪ q_j                 // subtract (q_hat * v) from u[j..]
      ↪ let q_hat_v = v.full_mul_u64(q_hat);          //
      ↪ u[j..] -= q_hat_v;          let c =
      ↪ Self::sub_slice(&mut u[j..], &q_hat_v[..n + 1]);
865      // D6.             // actually, q_hat == q_j + 1 and
      ↪ u[j..] has overflowed          // highly unlikely ~
      ↪ (1 / 2^63)          if c {          q_hat -=
      ↪ 1;                  // add v to u[j..]
      ↪ let c = Self::add_slice(&mut u[j..], &v.0[..n]);
      ↪ u[j + n] = u[j + n].wrapping_add(u64::from(c));
      ↪ }
866      // D5.             q.0[j] = q_hat;          }
867      // D8.             let remainder = Self::full_shr(u, shift);
868      (q, remainder)      }
869

```

- Call Stack

```

1  /home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-
  ↪ 0.9.1/src/uint.rs

```

- description:

Description of the bug here.

- link:

GitHub Link to be added.

- alleviation:

Some alleviation steps here.

Issue: INT_CVE_12: IntegerCve - Overflow

Category	Severity	Status
Integer Overflow	Critical	UnResolved

• Location

/home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-0.9.1/src/uint.rs:928:20: 928:25

928 j + n

929

• Code Context

/home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-0.9.1/src/uint.rs:859:4: 939:5

```

859 fn div_mod_knuth(self, mut v: Self, n: usize, m: usize) -> (Self, Self) {
    ↪ debug_assert!(self.bits() >= v.bits() && !v.fits_word());
    ↪ debug_assert!(n + m <= $n_words);           // D1.           // Make
    ↪ sure 64th bit in v's highest word is set.    // If we shift both
    ↪ self and v, it won't affect the quotient     // and the
    ↪ remainder will only need to be shifted back. let shift = v.0[n
    ↪ - 1].leading_zeros(); v <=< shift;           // u will store
    ↪ the remainder (shifted) let mut u = self.full_shl(shift);
860     // quotient let mut q = Self::zero(); let
    ↪ v_n_1 = v.0[n - 1]; let v_n_2 = v.0[n - 2];
861     // D2. D7. // iterate from m downto 0 for j
    ↪ in (0..=m).rev() { let u_jn = u[j + n];
862     // D3. // q_hat is our guess for the j-th
    ↪ quotient digit // q_hat = min(b - 1,
    ↪ (u_{j+n} * b + u_{j+n-1}) / v_{n-1}) // b
    ↪ = 1 << WORD_BITS // Theorem B: q_hat >=
    ↪ q_j >= q_hat - 2 let mut q_hat = if u_jn <
    ↪ v_n_1 { let (mut q_hat, mut r_hat) =
    ↪ Self::div_mod_word(u_jn, u[j + n - 1], v_n_1);
    ↪ // this loop takes at most 2 iterations
    ↪ loop { // check if q_hat * v_{n-2}
    ↪ > b * r_hat + u_{j+n-2} let (hi,
    ↪ lo) = Self::split_u128(u128::from(q_hat) *
    ↪ u128::from(v_n_2)); if (hi, lo) <=
    ↪ (r_hat, u[j + n - 2]) { break;
    ↪ } // then iterate till it doesn't
    ↪ hold q_hat -= 1;
    ↪ let (new_r_hat, overflow) =
O2Lab VRust Team r_hat.overflowing_add(v_n_1); 32
    ↪ r_hat = new_r_hat; // if r_hat
    ↪ overflowed, we're done if overflow
    ↪ { break;
    ↪ } q_hat
    ↪ }

```



```

863      // ex. 20:          // since q_hat * v_{n-2} <= b *
      ↪ r_hat + u_{j+n-2},          // either q_hat == q_j,
      ↪ or q_hat == q_j + 1
864      // D4.             // let's assume optimistically q_hat ==
      ↪ q_j                 // subtract (q_hat * v) from u[j..]
      ↪ let q_hat_v = v.full_mul_u64(q_hat);          //
      ↪ u[j..] -= q_hat_v;          let c =
      ↪ Self::sub_slice(&mut u[j..], &q_hat_v[..n + 1]);
865      // D6.             // actually, q_hat == q_j + 1 and
      ↪ u[j..] has overflowed          // highly unlikely ~
      ↪ (1 / 2^63)          if c {          q_hat -=
      ↪ 1;                  // add v to u[j..]
      ↪ let c = Self::add_slice(&mut u[j..], &v.0[..n]);
      ↪ u[j + n] = u[j + n].wrapping_add(u64::from(c));
      ↪ }
866      // D5.             q.0[j] = q_hat;          }
867      // D8.             let remainder = Self::full_shr(u, shift);
868      (q, remainder)      }
869

```

- Call Stack

```

1  /home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-
  ↪ 0.9.1/src/uint.rs

```

- description:

Description of the bug here.

- link:

GitHub Link to be added.

- alleviation:

Some alleviation steps here.

Issue: INT_CVE_13: IntegerCve - Overflow

Category	Severity	Status
Integer Overflow	Critical	UnResolved

• Location

/home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-0.9.1/src/uint.rs:928:9: 928:14

928 j + n

929

• Code Context

/home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-0.9.1/src/uint.rs:859:4: 939:5

```

859 fn div_mod_knuth(self, mut v: Self, n: usize, m: usize) -> (Self, Self) {
    ↪ debug_assert!(self.bits() >= v.bits() && !v.fits_word());
    ↪ debug_assert!(n + m <= $n_words);           // D1.           // Make
    ↪ sure 64th bit in v's highest word is set.    // If we shift both
    ↪ self and v, it won't affect the quotient     // and the
    ↪ remainder will only need to be shifted back. let shift = v.0[n
    ↪ - 1].leading_zeros();          v <=< shift;    // u will store
    ↪ the remainder (shifted)          let mut u = self.full_shl(shift);
860         // quotient          let mut q = Self::zero();          let
    ↪ v_n_1 = v.0[n - 1];          let v_n_2 = v.0[n - 2];
861     // D2. D7.           // iterate from m downto 0          for j
    ↪ in (0..=m).rev() {          let u_jn = u[j + n];
862     // D3.           // q_hat is our guess for the j-th
    ↪ quotient digit          // q_hat = min(b - 1,
    ↪ (u_{j+n} * b + u_{j+n-1}) / v_{n-1})          // b
    ↪ = 1 << WORD_BITS          // Theorem B: q_hat >=
    ↪ q_j >= q_hat - 2          let mut q_hat = if u_jn <
    ↪ v_n_1 {          let (mut q_hat, mut r_hat) =
    ↪ Self::div_mod_word(u_jn, u[j + n - 1], v_n_1);
    ↪ // this loop takes at most 2 iterations
    ↪ loop {          // check if q_hat * v_{n-2}
    ↪ > b * r_hat + u_{j+n-2}          let (hi,
    ↪ lo) = Self::split_u128(u128::from(q_hat) *
    ↪ u128::from(v_n_2));          if (hi, lo) <=
    ↪ (r_hat, u[j + n - 2]) {          break;
    ↪ }          // then iterate till it doesn't
    ↪ hold          q_hat -= 1;
    ↪ let (new_r_hat, overflow) =
O2Lab VRust Team          r_hat.overflowing_add(v_n_1);
    ↪ r_hat = new_r_hat;          // if r_hat
    ↪ overflowed, we're done          if overflow
    ↪ {          break;
    ↪ }          }          q_hat
    ↪ }          // (q_hat, r_hat) = (q_hat, r_hat)

```

```

863      // ex. 20:          // since q_hat * v_{n-2} <= b *
      ↪ r_hat + u_{j+n-2},          // either q_hat == q_j,
      ↪ or q_hat == q_j + 1
864      // D4.             // let's assume optimistically q_hat ==
      ↪ q_j                 // subtract (q_hat * v) from u[j..]
      ↪ let q_hat_v = v.full_mul_u64(q_hat);          //
      ↪ u[j..] -= q_hat_v;          let c =
      ↪ Self::sub_slice(&mut u[j..], &q_hat_v[..n + 1]);
865      // D6.             // actually, q_hat == q_j + 1 and
      ↪ u[j..] has overflowed          // highly unlikely ~
      ↪ (1 / 2^63)          if c {          q_hat -=
      ↪ 1;                  // add v to u[j..]
      ↪ let c = Self::add_slice(&mut u[j..], &v.0[..n]);
      ↪ u[j + n] = u[j + n].wrapping_add(u64::from(c));
      ↪ }
866      // D5.             q.0[j] = q_hat;          }
867      // D8.             let remainder = Self::full_shr(u, shift);
868      (q, remainder)      }
869

```

- Call Stack

```

1  /home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-
  ↪ 0.9.1/src/uint.rs

```

- description:

Description of the bug here.

- link:

GitHub Link to be added.

- alleviation:

Some alleviation steps here.

Issue: CHK_CVE_7: MissingCheckerCve - is_signer

Category	Severity	Status
Captured Signer Check	Informational	Resolved

- Location

programs/clearing_house/src/instructions.rs:122:10: 122:18

```
122 Accounts
123
```

- Call Stack

```
1 <instructions::InitializeMarket<'info> as
  ↳ anchor_lang::Accounts<'info>>::try_accounts
```

- description:

Captured is_signer check for function: <instructions::InitializeMarket<'info> as anchor_lang::Accounts<'info>>::try_accounts
We captured an is_signer check for variable: ::try_accounts

- link:

<https://github.com/parasol-aser/vrust/blob/yifei/patterns/01/README.md>

- alleviation:

Nothing needs to be done.

Issue: INT_CVE_14: IntegerCve - Overflow

Category	Severity	Status
Integer Overflow	Critical	UnResolved

- Location

/home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-0.9.1/src/uint.rs:1250:37:1250:58

```
1250 a as u128 * b as u128
1251
```

- Code Context

/home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-0.9.1/src/uint.rs:1249:4: 1252:5

```
1249 const fn mul_u64(a: u64, b: u64, carry: u64) -> (u64, u64) { let
↳ (hi, lo) = Self::split_u128(a as u128 * b as u128 + carry as u128);
↳ (lo, hi) }
1250
```

- Call Stack

```
1 /home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-
↳ 0.9.1/src/uint.rs
```

- description:

Description of the bug here.

- link:

GitHub Link to be added.

- alleviation:

Some alleviation steps here.

Issue: INT_CVE_15: IntegerCve - Overflow

Category	Severity	Status
Integer Overflow	Critical	UnResolved

- Location

/home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-0.9.1/src/uint.rs:1250:37:1250:74

```
1250 a as u128 * b as u128 + carry as u128
1251
```

- Code Context

/home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-0.9.1/src/uint.rs:1249:4: 1252:5

```
1249 const fn mul_u64(a: u64, b: u64, carry: u64) -> (u64, u64) { let
↳ (hi, lo) = Self::split_u128(a as u128 * b as u128 + carry as u128);
↳ (lo, hi) }
1250
```

- Call Stack

```
1 /home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-
↳ 0.9.1/src/uint.rs
```

- description:

Description of the bug here.

- link:

GitHub Link to be added.

- alleviation:

Some alleviation steps here.

Issue: INT_CVE_16: IntegerCve - Overflow

Category	Severity	Status
Integer Overflow	Critical	UnResolved

• Location

/home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-0.9.1/src/uint.rs:879:19: 879:24

```
879 j + n
880
```

• Code Context

/home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-0.9.1/src/uint.rs:859:4: 939:5

```
859 fn div_mod_knuth(self, mut v: Self, n: usize, m: usize) -> (Self, Self) {
    ↪ debug_assert!(self.bits() >= v.bits() && !v.fits_word());
    ↪ debug_assert!(n + m <= $n_words);           // D1.           // Make
    ↪ sure 64th bit in v's highest word is set.    // If we shift both
    ↪ self and v, it won't affect the quotient     // and the
    ↪ remainder will only need to be shifted back. let shift = v.0[n
    ↪ - 1].leading_zeros(); v <=< shift;           // u will store
    ↪ the remainder (shifted) let mut u = self.full_shl(shift);
860 // quotient let mut q = Self::zero(); let
    ↪ v_n_1 = v.0[n - 1]; let v_n_2 = v.0[n - 2];
861 // D2. D7. // iterate from m downto 0 for j
    ↪ in (0..=m).rev() { let u_jn = u[j + n];
862 // D3. // q_hat is our guess for the j-th
    ↪ quotient digit // q_hat = min(b - 1,
    ↪ (u_{j+n} * b + u_{j+n-1}) / v_{n-1}) // b
    ↪ = 1 << WORD_BITS // Theorem B: q_hat >=
    ↪ q_j >= q_hat - 2 let mut q_hat = if u_jn <
    ↪ v_n_1 { let (mut q_hat, mut r_hat) =
    ↪ Self::div_mod_word(u_jn, u[j + n - 1], v_n_1);
    ↪ // this loop takes at most 2 iterations
    ↪ loop { // check if q_hat * v_{n-2}
    ↪ > b * r_hat + u_{j+n-2} let (hi,
    ↪ lo) = Self::split_u128(u128::from(q_hat) *
    ↪ u128::from(v_n_2)); if (hi, lo) <=
    ↪ (r_hat, u[j + n - 2]) { break;
    ↪ } // then iterate till it doesn't
    ↪ hold q_hat -= 1;
    ↪ let (new_r_hat, overflow) =
O2Lab VRust Team r_hat.overflowing_add(v_n_1); 39
    ↪ r_hat = new_r_hat; // if r_hat
    ↪ overflowed, we're done if overflow
    ↪ { break;
    ↪ } q_hat
    ↪ } }
```

```

863      // ex. 20:          // since q_hat * v_{n-2} <= b *
      ↪ r_hat + u_{j+n-2},          // either q_hat == q_j,
      ↪ or q_hat == q_j + 1
864      // D4.             // let's assume optimistically q_hat ==
      ↪ q_j                 // subtract (q_hat * v) from u[j..]
      ↪ let q_hat_v = v.full_mul_u64(q_hat);          //
      ↪ u[j..] -= q_hat_v;          let c =
      ↪ Self::sub_slice(&mut u[j..], &q_hat_v[..n + 1]);
865      // D6.             // actually, q_hat == q_j + 1 and
      ↪ u[j..] has overflowed          // highly unlikely ~
      ↪ (1 / 2^63)          if c {          q_hat -=
      ↪ 1;                  // add v to u[j..]
      ↪ let c = Self::add_slice(&mut u[j..], &v.0[..n]);
      ↪ u[j + n] = u[j + n].wrapping_add(u64::from(c));
      ↪ }
866      // D5.             q.0[j] = q_hat;          }
867      // D8.             let remainder = Self::full_shr(u, shift);
868      (q, remainder)      }
869

```

- Call Stack

```

1  /home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-
  ↪ 0.9.1/src/uint.rs

```

- description:

Description of the bug here.

- link:

GitHub Link to be added.

- alleviation:

Some alleviation steps here.

Issue: INT_CVE_17: IntegerCve - Overflow

Category	Severity	Status
Integer Overflow	Critical	UnResolved

- Location

```
/home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-0.9.1/src/uint.rs:887:63: 887:68
```

887 $j + n$

888

- Code Context

```
/home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-0.9.1/src/uint.rs:859:4: 939:5
```

```

859 fn div_mod_knuth(self, mut v: Self, n: usize, m: usize) -> (Self, Self) {
    ↪ debug_assert!(self.bits() >= v.bits() && !v.fits_word());
    ↪ debug_assert!(n + m <= $n_words); // D1. // Make
    ↪ sure 64th bit in v's highest word is set. // If we shift both
    ↪ self and v, it won't affect the quotient // and the
    ↪ remainder will only need to be shifted back. let shift = v.0[n
    ↪ - 1].leading_zeros(); v <<= shift; // u will store
    ↪ the remainder (shifted) let mut u = self.full_shl(shift);
860 // quotient let mut q = Self::zero(); let
    ↪ v_n_1 = v.0[n - 1]; let v_n_2 = v.0[n - 2];
861 // D2. D7. // iterate from m downto 0 for j
    ↪ in (0..=m).rev() { let u_jn = u[j + n];
862 // D3. // q_hat is our guess for the j-th
    ↪ quotient digit // q_hat = min(b - 1,
    ↪ (u_{j+n} * b + u_{j+n-1}) / v_{n-1}) // b
    ↪ = 1 << WORD_BITS // Theorem B: q_hat >=
    ↪ q_j >= q_hat - 2 let mut q_hat = if u_jn <
    ↪ v_n_1 { let (mut q_hat, mut r_hat) =
    ↪ Self::div_mod_word(u_jn, u[j + n - 1], v_n_1);
    ↪ // this loop takes at most 2 iterations
    ↪ loop { // check if q_hat * v_{n-2}
    ↪ > b * r_hat + u_{j+n-2} let (hi,
    ↪ lo) = Self::split_u128(u128::from(q_hat) *
    ↪ u128::from(v_n_2)); if (hi, lo) <=
    ↪ (r_hat, u[j + n - 2]) { break;
    ↪ } // then iterate till it doesn't
    ↪ hold q_hat -= 1;
    ↪ let (new_r_hat, overflow) =
863 r_hat.overflowing_add(v_n_1);
    ↪ r_hat = new_r_hat; // if r_hat
    ↪ overflowed, we're done if overflow
    ↪ { break;
    ↪ } q_hat

```

```

863      // ex. 20:          // since q_hat * v_{n-2} <= b *
      ↪ r_hat + u_{j+n-2},          // either q_hat == q_j,
      ↪ or q_hat == q_j + 1
864      // D4.             // let's assume optimistically q_hat ==
      ↪ q_j                 // subtract (q_hat * v) from u[j..]
      ↪ let q_hat_v = v.full_mul_u64(q_hat);          //
      ↪ u[j..] -= q_hat_v;          let c =
      ↪ Self::sub_slice(&mut u[j..], &q_hat_v[..n + 1]);
865      // D6.             // actually, q_hat == q_j + 1 and
      ↪ u[j..] has overflowed          // highly unlikely ~
      ↪ (1 / 2^63)          if c {          q_hat -=
      ↪ 1;                  // add v to u[j..]
      ↪ let c = Self::add_slice(&mut u[j..], &v.0[..n]);
      ↪ u[j + n] = u[j + n].wrapping_add(u64::from(c));
      ↪ }
866      // D5.             q.0[j] = q_hat;          }
867      // D8.             let remainder = Self::full_shr(u, shift);
868      (q, remainder)      }
869

```

- Call Stack

```

1  /home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-
  ↪ 0.9.1/src/uint.rs

```

- description:

Description of the bug here.

- link:

GitHub Link to be added.

- alleviation:

Some alleviation steps here.

Issue: INT_CVE_18: IntegerCve - Overflow

Category	Severity	Status
Integer Overflow	Critical	UnResolved

• Location

/home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-0.9.1/src/uint.rs:892:33: 892:38

```
892 j + n
893
```

• Code Context

/home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-0.9.1/src/uint.rs:859:4: 939:5

```
859 fn div_mod_knuth(self, mut v: Self, n: usize, m: usize) -> (Self, Self) {
    ↪ debug_assert!(self.bits() >= v.bits() && !v.fits_word());
    ↪ debug_assert!(n + m <= $n_words);           // D1.           // Make
    ↪ sure 64th bit in v's highest word is set.    // If we shift both
    ↪ self and v, it won't affect the quotient     // and the
    ↪ remainder will only need to be shifted back. let shift = v.0[n
    ↪ - 1].leading_zeros(); v <=< shift;           // u will store
    ↪ the remainder (shifted) let mut u = self.full_shl(shift);
860 // quotient let mut q = Self::zero(); let
    ↪ v_n_1 = v.0[n - 1]; let v_n_2 = v.0[n - 2];
861 // D2. D7. // iterate from m downto 0 for j
    ↪ in (0..=m).rev() { let u_jn = u[j + n];
862 // D3. // q_hat is our guess for the j-th
    ↪ quotient digit // q_hat = min(b - 1,
    ↪ (u_{j+n} * b + u_{j+n-1}) / v_{n-1}) // b
    ↪ = 1 << WORD_BITS // Theorem B: q_hat >=
    ↪ q_j >= q_hat - 2 let mut q_hat = if u_jn <
    ↪ v_n_1 { let (mut q_hat, mut r_hat) =
    ↪ Self::div_mod_word(u_jn, u[j + n - 1], v_n_1);
    ↪ // this loop takes at most 2 iterations
    ↪ loop { // check if q_hat * v_{n-2}
    ↪ > b * r_hat + u_{j+n-2} let (hi,
    ↪ lo) = Self::split_u128(u128::from(q_hat) *
    ↪ u128::from(v_n_2)); if (hi, lo) <=
    ↪ (r_hat, u[j + n - 2]) { break;
    ↪ } // then iterate till it doesn't
    ↪ hold q_hat -= 1;
    ↪ let (new_r_hat, overflow) =
O2Lab VRust Team r_hat.overflowing_add(v_n_1); 43
    ↪ r_hat = new_r_hat; // if r_hat
    ↪ overflowed, we're done if overflow
    ↪ { break;
    ↪ } q_hat
    ↪ } // then iterate till it doesn't
```

```

863      // ex. 20:          // since q_hat * v_{n-2} <= b *
      ↪ r_hat + u_{j+n-2},          // either q_hat == q_j,
      ↪ or q_hat == q_j + 1
864      // D4.             // let's assume optimistically q_hat ==
      ↪ q_j                 // subtract (q_hat * v) from u[j..]
      ↪ let q_hat_v = v.full_mul_u64(q_hat);          //
      ↪ u[j..] -= q_hat_v;          let c =
      ↪ Self::sub_slice(&mut u[j..], &q_hat_v[..n + 1]);
865      // D6.             // actually, q_hat == q_j + 1 and
      ↪ u[j..] has overflowed          // highly unlikely ~
      ↪ (1 / 2^63)          if c {          q_hat -=
      ↪ 1;                  // add v to u[j..]
      ↪ let c = Self::add_slice(&mut u[j..], &v.0[..n]);
      ↪ u[j + n] = u[j + n].wrapping_add(u64::from(c));
      ↪ }
866      // D5.             q.0[j] = q_hat;          }
867      // D8.             let remainder = Self::full_shr(u, shift);
868      (q, remainder)      }
869

```

- Call Stack

```

1  /home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-
   ↪ 0.9.1/src/uint.rs

```

- description:

Description of the bug here.

- link:

GitHub Link to be added.

- alleviation:

Some alleviation steps here.

Issue: INT_CVE_19: IntegerCve - Overflow

Category	Severity	Status
Integer Overflow	Critical	UnResolved

• Location

/home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-0.9.1/src/uint.rs:928:20: 928:25

928 j + n

929

• Code Context

/home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-0.9.1/src/uint.rs:859:4: 939:5

```

859 fn div_mod_knuth(self, mut v: Self, n: usize, m: usize) -> (Self, Self) {
    ↪ debug_assert!(self.bits() >= v.bits() && !v.fits_word());
    ↪ debug_assert!(n + m <= $n_words);           // D1.           // Make
    ↪ sure 64th bit in v's highest word is set.    // If we shift both
    ↪ self and v, it won't affect the quotient      // and the
    ↪ remainder will only need to be shifted back.  let shift = v.0[n
    ↪ - 1].leading_zeros();          v <=< shift;    // u will store
    ↪ the remainder (shifted)          let mut u = self.full_shl(shift);
860         // quotient          let mut q = Self::zero();          let
    ↪ v_n_1 = v.0[n - 1];          let v_n_2 = v.0[n - 2];
861     // D2. D7.           // iterate from m downto 0          for j
    ↪ in (0..=m).rev() {          let u_jn = u[j + n];
862     // D3.           // q_hat is our guess for the j-th
    ↪ quotient digit          // q_hat = min(b - 1,
    ↪ (u_{j+n} * b + u_{j+n-1}) / v_{n-1})          // b
    ↪ = 1 << WORD_BITS          // Theorem B: q_hat >=
    ↪ q_j >= q_hat - 2          let mut q_hat = if u_jn <
    ↪ v_n_1 {          let (mut q_hat, mut r_hat) =
    ↪ Self::div_mod_word(u_jn, u[j + n - 1], v_n_1);
    ↪ // this loop takes at most 2 iterations
    ↪ loop {          // check if q_hat * v_{n-2}
    ↪ > b * r_hat + u_{j+n-2}          let (hi,
    ↪ lo) = Self::split_u128(u128::from(q_hat) *
    ↪ u128::from(v_n_2));          if (hi, lo) <=
    ↪ (r_hat, u[j + n - 2]) {          break;
    ↪ }          // then iterate till it doesn't
    ↪ hold          q_hat -= 1;
    ↪ let (new_r_hat, overflow) =
O2Lab VRust Team          r_hat.overflowing_add(v_n_1);
    ↪ r_hat = new_r_hat;          // if r_hat
    ↪ overflowed, we're done          if overflow
    ↪ {          break;
    ↪ }          }          q_hat
    ↪ }          // (q_hat, r_hat) = (q_hat, r_hat)

```

```

863      // ex. 20:          // since q_hat * v_{n-2} <= b *
      ↪ r_hat + u_{j+n-2},          // either q_hat == q_j,
      ↪ or q_hat == q_j + 1
864      // D4.             // let's assume optimistically q_hat ==
      ↪ q_j                 // subtract (q_hat * v) from u[j..]
      ↪ let q_hat_v = v.full_mul_u64(q_hat);          //
      ↪ u[j..] -= q_hat_v;          let c =
      ↪ Self::sub_slice(&mut u[j..], &q_hat_v[..n + 1]);
865      // D6.             // actually, q_hat == q_j + 1 and
      ↪ u[j..] has overflowed          // highly unlikely ~
      ↪ (1 / 2^63)          if c {          q_hat -=
      ↪ 1;                  // add v to u[j..]
      ↪ let c = Self::add_slice(&mut u[j..], &v.0[..n]);
      ↪ u[j + n] = u[j + n].wrapping_add(u64::from(c));
      ↪ }
866      // D5.             q.0[j] = q_hat;          }
867      // D8.             let remainder = Self::full_shr(u, shift);
868      (q, remainder)      }
869

```

- Call Stack

```

1  /home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-
  ↪ 0.9.1/src/uint.rs

```

- description:

Description of the bug here.

- link:

GitHub Link to be added.

- alleviation:

Some alleviation steps here.

Issue: INT_CVE_20: IntegerCve - Overflow

Category	Severity	Status
Integer Overflow	Critical	UnResolved

• Location

/home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-0.9.1/src/uint.rs:928:9: 928:14

928 j + n

929

• Code Context

/home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-0.9.1/src/uint.rs:859:4: 939:5

```

859 fn div_mod_knuth(self, mut v: Self, n: usize, m: usize) -> (Self, Self) {
    ↪ debug_assert!(self.bits() >= v.bits() && !v.fits_word());
    ↪ debug_assert!(n + m <= $n_words);           // D1.           // Make
    ↪ sure 64th bit in v's highest word is set.    // If we shift both
    ↪ self and v, it won't affect the quotient     // and the
    ↪ remainder will only need to be shifted back. let shift = v.0[n
    ↪ - 1].leading_zeros();          v <= shift;    // u will store
    ↪ the remainder (shifted)         let mut u = self.full_shl(shift);
860         // quotient         let mut q = Self::zero();         let
    ↪ v_n_1 = v.0[n - 1];         let v_n_2 = v.0[n - 2];
861     // D2. D7.           // iterate from m downto 0         for j
    ↪ in (0..=m).rev() {         let u_jn = u[j + n];
862     // D3.           // q_hat is our guess for the j-th
    ↪ quotient digit           // q_hat = min(b - 1,
    ↪ (u_{j+n} * b + u_{j+n-1}) / v_{n-1})           // b
    ↪ = 1 << WORD_BITS           // Theorem B: q_hat >=
    ↪ q_j >= q_hat - 2         let mut q_hat = if u_jn <
    ↪ v_n_1 {         let (mut q_hat, mut r_hat) =
    ↪ Self::div_mod_word(u_jn, u[j + n - 1], v_n_1);
    ↪ // this loop takes at most 2 iterations
    ↪ loop {           // check if q_hat * v_{n-2}
    ↪ > b * r_hat + u_{j+n-2}         let (hi,
    ↪ lo) = Self::split_u128(u128::from(q_hat) *
    ↪ u128::from(v_n_2));           if (hi, lo) <=
    ↪ (r_hat, u[j + n - 2]) {         break;
    ↪ }           // then iterate till it doesn't
    ↪ hold         q_hat -= 1;
    ↪ let (new_r_hat, overflow) =
O2Lab VRust Team         r_hat.overflowing_add(v_n_1);
    ↪ r_hat = new_r_hat;           // if r_hat
    ↪ overflowed, we're done         if overflow
    ↪ {         break;
    ↪ }         }         q_hat
    ↪ }         }         // (q_hat, r_hat) =

```

```

863      // ex. 20:          // since q_hat * v_{n-2} <= b *
      ↪ r_hat + u_{j+n-2},          // either q_hat == q_j,
      ↪ or q_hat == q_j + 1
864      // D4.             // let's assume optimistically q_hat ==
      ↪ q_j                 // subtract (q_hat * v) from u[j..]
      ↪ let q_hat_v = v.full_mul_u64(q_hat);          //
      ↪ u[j..] -= q_hat_v;          let c =
      ↪ Self::sub_slice(&mut u[j..], &q_hat_v[..n + 1]);
865      // D6.             // actually, q_hat == q_j + 1 and
      ↪ u[j..] has overflowed          // highly unlikely ~
      ↪ (1 / 2^63)          if c {          q_hat -=
      ↪ 1;                  // add v to u[j..]
      ↪ let c = Self::add_slice(&mut u[j..], &v.0[..n]);
      ↪ u[j + n] = u[j + n].wrapping_add(u64::from(c));
      ↪ }
866      // D5.             q.0[j] = q_hat;          }
867      // D8.             let remainder = Self::full_shr(u, shift);
868      (q, remainder)      }
869

```

- Call Stack

```

1  /home/tien/.cargo/registry/src/github.com-1ecc6299db9ec823/uint-
  ↪ 0.9.1/src/uint.rs

```

- description:

Description of the bug here.

- link:

GitHub Link to be added.

- alleviation:

Some alleviation steps here.

Issue: CHK_CVE_8: MissingCheckerCve - is_signer

Category	Severity	Status
Captured Signer Check	Informational	Resolved

- Location

programs/clearing_house/src/instructions.rs:99:10: 99:18

```
99 Accounts
100
```

- Call Stack

```
1 <instructions::DeleteUser<'info> as
  ↳ anchor_lang::Accounts<'info>>::try_accounts
```

- description:

Captured is_signer check for function: <instructions::DeleteUser<'info> as anchor_lang::Accounts<'info>>::try_accounts
We captured an is_signer check for variable: ::try_accounts

- link:

<https://github.com/parasol-aser/vrust/blob/yifei/patterns/01/README.md>

- alleviation:

Nothing needs to be done.

Issue: CHK_CVE_9: MissingCheckerCve - is_signer

Category	Severity	Status
Captured Signer Check	Informational	Resolved

- Location

programs/clearing_house/src/instructions.rs:277:10: 277:18

```
277 Accounts
278
```

- Call Stack

```
1 <instructions::WithdrawFromInsuranceVaultToMarket<'info> as
  ↳ anchor_lang::Accounts<'info>>::try_accounts
```

- description:

Captured is_signer check for function: <instructions::WithdrawFromInsuranceVaultToMarket<'info> as anchor_lang::Accounts<'info>>::try_accounts We captured an is_signer check for variable: ::try_accounts

- link:

<https://github.com/parasol-aser/vrust/blob/yifei/patterns/01/README.md>

- alleviation:

Nothing needs to be done.

Issue: CHK_CVE_10: MissingCheckerCve - is_signer

Category	Severity	Status
Captured Signer Check	Informational	Resolved

- Location

programs/clearing_house/src/instructions.rs:177:10: 177:18

```
177 Accounts
178
```

- Call Stack

```
1 <instructions::WithdrawCollateral<'info> as
  ↳ anchor_lang::Accounts<'info>>::try_accounts
```

- description:

Captured is_signer check for function: <instructions::WithdrawCollateral<'info> as anchor_lang::Accounts<'info>>::try_a
We captured an is_signer check for variable: ::try_accounts

- link:

<https://github.com/parasol-aser/vrust/blob/yifei/patterns/01/README.md>

- alleviation:

Nothing needs to be done.

Issue: CHK_CVE_11: MissingCheckerCve - is_signer

Category	Severity	Status
Captured Signer Check	Informational	Resolved

- Location

programs/clearing_house/src/instructions.rs:256:10: 256:18

256 Accounts
257

- Call Stack

```
1 <instructions::WithdrawFromInsuranceVault<'info> as
  ↳ anchor_lang::Accounts<'info>::try_accounts
```

- description:

Captured is_signer check for function: <instructions::WithdrawFromInsuranceVault<'info> as anchor_lang::Accounts<'info>::try_accounts We captured an is_signer check for variable: ::try_accounts

- link:

<https://github.com/parasol-aser/vrust/blob/yifei/patterns/01/README.md>

- alleviation:

Nothing needs to be done.

Issue: INT_CVE_21: IntegerCve - Overflow

Category	Severity	Status
Integer Overflow	Critical	UnResolved

- Location

programs/clearing_house/src/lib.rs:27:1: 27:11

```
27  #[program]
```

```
28
```

- Code Context

programs/clearing_house/src/lib.rs:27:1: 27:11

```
27  #[program]
```

```
28
```

- Call Stack

```
1  programs/clearing_house/src/lib.rs
```

- description:

Description of the bug here.

- link:

GitHub Link to be added.

- alleviation:

Some alleviation steps here.

Issue: CHK_CVE_12: MissingCheckerCve - is_signer

Category	Severity	Status
Captured Signer Check	Informational	Resolved

- Location

programs/clearing_house/src/instructions.rs:511:10: 511:18

```
511 Accounts
512
```

- Call Stack

```
1 <instructions::MoveAMMPrice<'info> as
  ↳ anchor_lang::Accounts<'info>>::try_accounts
```

- description:

Captured is_signer check for function: <instructions::MoveAMMPrice<'info> as anchor_lang::Accounts<'info>>::try_accounts
We captured an is_signer check for variable: ::try_accounts

- link:

<https://github.com/parasol-aser/vrust/blob/yifei/patterns/01/README.md>

- alleviation:

Nothing needs to be done.

Issue: CHK_CVE_13: MissingCheckerCve - is_signer

Category	Severity	Status
Captured Signer Check	Informational	Resolved

- Location

programs/clearing_house/src/instructions.rs:526:10: 526:18

526
527

Accounts

- Call Stack

```
1 <instructions::AdminUpdateState<'info> as  
  ↪ anchor_lang::Accounts<'info>>::try_accounts
```

- description:

Captured is_signer check for function: <instructions::AdminUpdateState<'info> as anchor_lang::Accounts<'info>>::try_a
We captured an is_signer check for variable: ::try_accounts

- link:

<https://github.com/parasol-aser/vrust/blob/yifei/patterns/01/README.md>

- alleviation:

Nothing needs to be done.

Issue: CHK_CVE_14: MissingCheckerCve - is_signer

Category	Severity	Status
Captured Signer Check	Informational	Resolved

- Location

programs/clearing_house/src/instructions.rs:552:10: 552:18

```
552 Accounts
553
```

- Call Stack

```
1 <instructions::AdminUpdateMarket<'info> as
  ↳ anchor_lang::Accounts<'info>>::try_accounts
```

- description:

Captured is_signer check for function: <instructions::AdminUpdateMarket<'info> as anchor_lang::Accounts<'info>>::try_accounts
We captured an is_signer check for variable: ::try_accounts

- link:

<https://github.com/parasol-aser/vrust/blob/yifei/patterns/01/README.md>

- alleviation:

Nothing needs to be done.

Issue: CHK_CVE_15: MissingCheckerCve - is_signer

Category	Severity	Status
Captured Signer Check	Informational	Resolved

- Location

programs/clearing_house/src/instructions.rs:56:10: 56:18

56 Accounts

57

- Call Stack

```
1 <instructions::InitializeHistory<'info> as  
  ↪ anchor_lang::Accounts<'info>>::try_accounts
```

- description:

Captured is_signer check for function: <instructions::InitializeHistory<'info> as anchor_lang::Accounts<'info>>::try_accounts
We captured an is_signer check for variable: ::try_accounts

- link:

<https://github.com/parasol-aser/vrust/blob/yifei/patterns/01/README.md>

- alleviation:

Nothing needs to be done.

Issue: CHK_CVE_16: MissingCheckerCve - is_signer

Category	Severity	Status
Captured Signer Check	Informational	Resolved

- Location

programs/clearing_house/src/instructions.rs:491:10: 491:18

491 Accounts

492

- Call Stack

```
1 <instructions::RepegCurve<'info> as  
  ↳ anchor_lang::Accounts<'info>>::try_accounts
```

- description:

Captured is_signer check for function: <instructions::RepegCurve<'info> as anchor_lang::Accounts<'info>>::try_account
We captured an is_signer check for variable: ::try_accounts

- link:

<https://github.com/parasol-aser/vrust/blob/yifei/patterns/01/README.md>

- alleviation:

Nothing needs to be done.

Issue: CHK_CVE_17: MissingCheckerCve - is_signer

Category	Severity	Status
Captured Signer Check	Informational	Resolved

- Location

programs/clearing_house/src/instructions.rs:352:10: 352:18

```
352 Accounts
353
```

- Call Stack

```
1 <instructions::ClosePosition<'info> as
  ↳ anchor_lang::Accounts<'info>>::try_accounts
```

- description:

Captured is_signer check for function: <instructions::ClosePosition<'info> as anchor_lang::Accounts<'info>>::try_accounts
We captured an is_signer check for variable: ::try_accounts

- link:

<https://github.com/parasol-aser/vrust/blob/yifei/patterns/01/README.md>

- alleviation:

Nothing needs to be done.

Appendix

Copied from <https://leaderboard.certik.io/projects/aave>

Finding Categories

Gas Optimization

Gas Optimization findings do not affect the functionality of the code but generate different, more optimal EVM opcodes resulting in a reduction on the total gas cost of a transaction.

Mathematical Operations

Mathematical Operation findings relate to mishandling of math formulas, such as overflows, incorrect operations etc.

Logical Issue

Logical Issue findings detail a fault in the logic of the linked code, such as an incorrect notion on how `block.timestamp` works.

Language Specific

Language Specific findings are issues that would only arise within Solidity, i.e. incorrect usage of `private` or `delete`.

Coding Style

Coding Style findings usually do not affect the generated byte-code but rather comment on how to make the codebase more legible and, as a result, easily maintainable.

Checksum Calculation Method

The “Checksum” field in the “Audit Scope” section is calculated as the SHA-256 (Secure Hash Algorithm 2 with digest size of 256 bits) digest of the content of each file hosted in the listed source repository under the specified commit.

The result is hexadecimal encoded and is the same as the output of the Linux “sha256sum” command against the target file.

Disclaimer

Copied from <https://leaderboard.certik.io/projects/aave>

This report is subject to the terms and conditions (including without limitation, description of services, confidentiality, disclaimer and limitation of liability) set forth in the Services Agreement, or the scope of services, and terms and conditions provided to you (“Customer” or the “Company”) in connection with the Agreement. This report provided in connection with the Services set forth in the Agreement shall be used by the Company only to the extent permitted under the terms and conditions set forth in the Agreement. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes, nor may copies be delivered to any other person other than the Company, without CertiK’s prior written consent in each instance.

This report is not, nor should be considered, an “endorsement” or “disapproval” of any particular project or team. This report is not, nor should be considered, an indication of the economics or value of any “product” or “asset” created by any team or project that contracts CertiK to perform a security assessment. This report does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors, business, business model or legal compliance.

This report should not be used in any way to make decisions around investment or involvement with any particular project. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology. Blockchain technology and cryptographic assets present a high level of ongoing risk. CertiK’s position is that each company and individual are responsible for their own due diligence and continuous security. CertiK’s goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies, and in no way claims any guarantee of security or functionality of the technology we agree to analyze. The assessment services provided by CertiK is subject to dependencies and under continuing development. You agree that your access and/or use, including but not limited to any services, reports, and materials, will be at your sole risk on an as-is, where-is, and as-available basis. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives, false negatives, and other unpredictable results. The services may access, and depend upon, multiple layers of third-parties.