// HALBORN

Solana Foundation - Vote Account Manager

Solana Program Security Audit

Prepared by: **Halborn**

Date of Engagement: **January 25th, 2023 - February 14th, 2023**

Visit: **Halborn.com**

DOCUMENT REVISION HISTORY

VERSION	MODIFICATION	DATE	AUTHOR
0.1	Document Creation	01/26/2023	Isabel Burruezo
0.2	Document Updates	02/10/2023	Isabel Burruezo
0.3	Final Draft	02/14/2023	Isabel Burruezo
0.4	Draft Review	02/14/2023	Piotr Cielas
0.5	Draft Review	02/14/2023	Gabi Urrutia
1.0	Remediation Plan	02/17/2023	Isabel Burruezo
1.1	Remediation Plan	02/20/2023	Piotr Cielas
1.2	Remediation Plan	02/20/2023	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 Vote Account Manager on-chain program implements a mechanism to manage a Solana vote account with better security and with additional features. It supports five authorities which control different aspects of the Vote Account, compared to two highly overlapped authorities with stock Vote Account Management. In addition, the owner of a vote account may optionally configure this program to enforce commission caps on the vote account.

Solana Foundation engaged Halborn to conduct a security audit on their Solana program, beginning on January 25th, 2023 and ending on February 14th, 2023 . The security assessment was scoped to the program provided in the vote-account-manager 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 a week and a half 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 smart contract security expert with advanced penetration testing and smart contract 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 an improvement to reduce risk probability and impact, which was successfully addressed by the Solana Foundation team.

1.3 TEST APPROACH & METHODOLOGY

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

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

- Research into the architecture, purpose, and use of the platform.
- Manual program source code review to identify business logic issues.
- Mapping out possible attack vectors
- Thorough assessment of safety and usage of critical Rust variables and functions in scope that could lead to arithmetic vulnerabilities.
- Finding bugs, detecting vulnerabilities in third-party dependencies (Semgrep)

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:

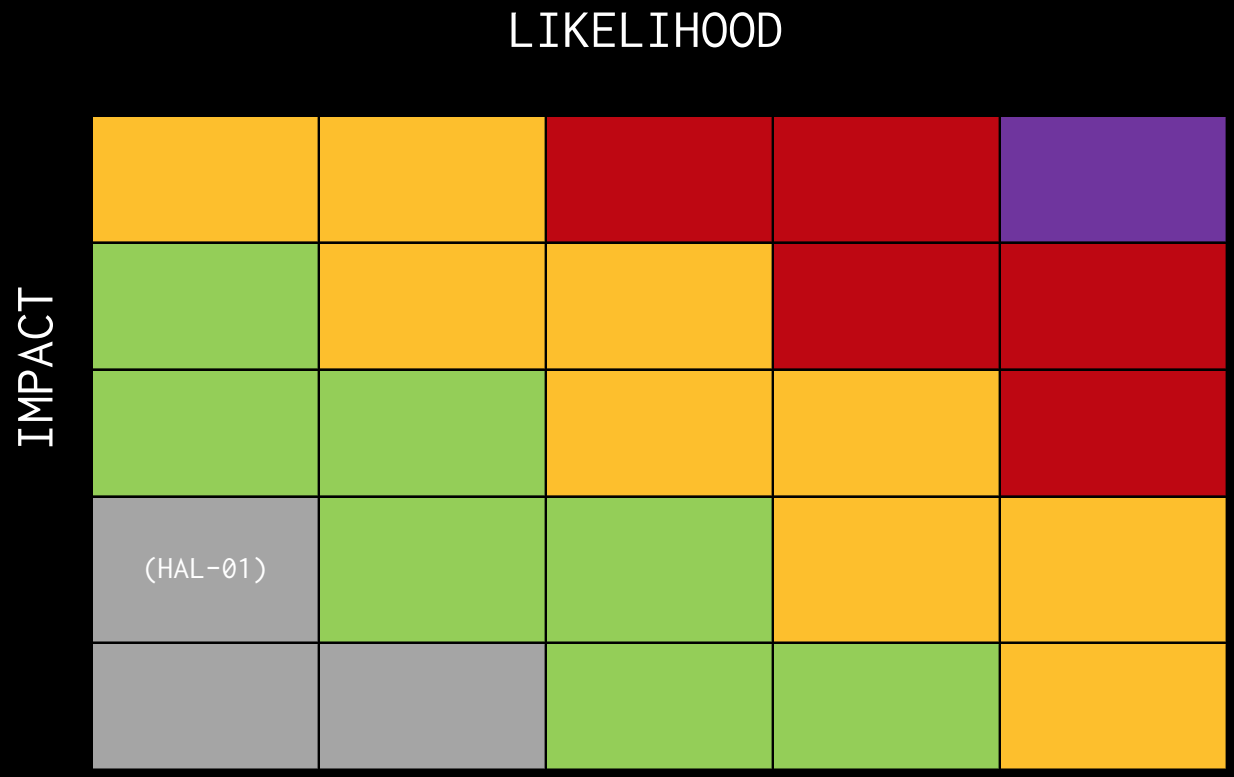
1. Vote Account Manager

- Repository: vote-account-manager
- Commit ID: ea88d5876062ddbfc83882f9af95ec32d8e94af6
- Programs in scope:
    1. vote-account-manager (vote-account-manager/program)

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

2. ASSESSMENT SUMMARY & FINDINGS OVERVIEW

CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL
0	0	0	0	1



SECURITY ANALYSIS	RISK LEVEL	REMEDIATION DATE
(HAL-01) CLOSED MANAGER ACCOUNT CAN BE REOPENED	Informational	SOLVED - 02/15/2023

FINDINGS & TECH DETAILS

3.1 (HAL-01) CLOSED MANAGER ACCOUNT CAN BE REOPENED - INFORMATIONAL

Description:

The Leave instruction can be called by the withdraw_authority only. It restores the withdraw authority of a vote account to its original value and transfers all lamports from the manager account to the recipient account, and deletes the manager account.

However, it is worth noting that this instruction does not purge manager_account data-it merely transfers the remaining lamports. The runtime deletes the account when the transaction is successfully executed, but until then, the manager account can be used and/pr recovered by simply transferring enough lamports to it from another account to cover the rent.

Code Location:

Listing 1: program/entrypoint.c (Lines 1141,1142)

```
1135 uint64_t ret = sol_invoke_signed(&instruction, params->ka, params
 ↳ ->ka_num, signer_seeds, 1);
1136     if (ret) {
1137         return ret;
1138     }
1139
1140     // Now return all lamports from the manager account to the
 ↳ recipient account, thus deleting the manager acount
1141     *(recipient_account->lamports) += *(manager_account->lamports)
 ↳ ;
1142     *(manager_account->lamports) = 0;
1143
1144     return 0;
```

Risk Level:

Likelihood - 1
Impact - 2

Recommendation:

It is recommended to purge properly the manager account zeroing its data.

Remediation Plan-->:

SOLVED The project team fixed this issue in commit:
- d65ac42f22e07ddf169fb05d1cca0727ac89261a

Now, manager account data is properly purged on close.

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 REMOVE VOTE ACCOUNT FROM MANAGEMENT BEFORE SETTING LEAVE EPOCH

Description:

If the Enter instruction is executed with commission caps enabled for a vote account, then it cannot be removed from management by the Vote Account Manager program until a **leave epoch** has been set. This restriction was tested to confirm it is safe, and no vulnerabilities were introduced.

Results:

No vulnerabilities were identified.

```
[**] Enter  Instruction
[+]   EnterInstructionData { instruction_index: 0, administrator: FkGNfeyQH7dVvfqTQi5rAt12puEER8RqUuEpquLwcY3R, use_commission_caps: true, max_commission: 50, max_commission_increase_per_epoch: 20 }
[2023-02-10T09:21:18.105880000Z DEBUG solana_runtime::message_processor::stable_log] Program vamp2RmRDCj5ZW8rGzU1vKh3NCudJcbW8M8ey7q1CXn invoke [1]
[2023-02-10T09:21:18.106700000Z DEBUG solana_runtime::message_processor::stable_log] Program 11111111111111111111111111111111 invoke [2]
[2023-02-10T09:21:18.106728000Z TRACE solana_runtime::system_instruction_processor] process_instruction: Transfer { lamports: 2804480 }
[2023-02-10T09:21:18.106748000Z DEBUG solana_runtime::message_processor::stable_log] Program 11111111111111111111111111111111 success
[2023-02-10T09:21:18.106936000Z DEBUG solana_runtime::message_processor::stable_log] Program 11111111111111111111111111111111 invoke [2]
[2023-02-10T09:21:18.106962000Z TRACE solana_runtime::system_instruction_processor] process_instruction: Allocate { space: 160 }
[2023-02-10T09:21:18.106983000Z DEBUG solana_runtime::message_processor::stable_log] Program 11111111111111111111111111111111 success
[2023-02-10T09:21:18.107126000Z DEBUG solana_runtime::message_processor::stable_log] Program 11111111111111111111111111111111 invoke [2]
[2023-02-10T09:21:18.107153000Z TRACE solana_runtime::system_instruction_processor] process_instruction: Assign { owner: vamp2RmRDCj5ZW8rGzU1vKh3NCudJcbW8M8ey7q1CXn }
[2023-02-10T09:21:18.107204000Z DEBUG solana_runtime::message_processor::stable_log] Program 11111111111111111111111111111111 success
[2023-02-10T09:21:18.107298000Z DEBUG solana_runtime::message_processor::stable_log] Program Vote111111111111111111111111111111111111111 invoke [2]
[2023-02-10T09:21:18.107689000Z DEBUG solana_runtime::message_processor::stable_log] Program Vote111111111111111111111111111111111111111 success
[2023-02-10T09:21:18.107764000Z DEBUG solana_runtime::message_processor::stable_log] Program vamp2RmRDCj5ZW8rGzU1vKh3NCudJcbW8M8ey7q1CXn consumed 6956 of 200000 compute units
[2023-02-10T09:21:18.107836000Z DEBUG solana_runtime::message_processor::stable_log] Program vamp2RmRDCj5ZW8rGzU1vKh3NCudJcbW8M8ey7q1CXn success
[**] Enter  Instruction done!
[+] Leave  Instruction
[2023-02-10T09:21:18.118933000Z DEBUG solana_runtime::message_processor::stable_log] Program vamp2RmRDCj5ZW8rGzU1vKh3NCudJcbW8M8ey7q1CXn invoke [1]
[2023-02-10T09:21:18.119432000Z DEBUG solana_runtime::message_processor::stable_log] Program vamp2RmRDCj5ZW8rGzU1vKh3NCudJcbW8M8ey7q1CXn consumed 2509 of 200000 compute units
[2023-02-10T09:21:18.119495000Z DEBUG solana_runtime::message_processor::stable_log] Program vamp2RmRDCj5ZW8rGzU1vKh3NCudJcbW8M8ey7q1CXn failed: custom program error: 0x3f2
```

The error got in hexadecimal corresponds to the following one:

Listing 2: program/entrypoint.c (Line 262)

```
261 // Attempt to leave when a leave epoch is required but has not
 ↳ been set
262 Error_LeaveEpochNotSet                       = 1010,
```

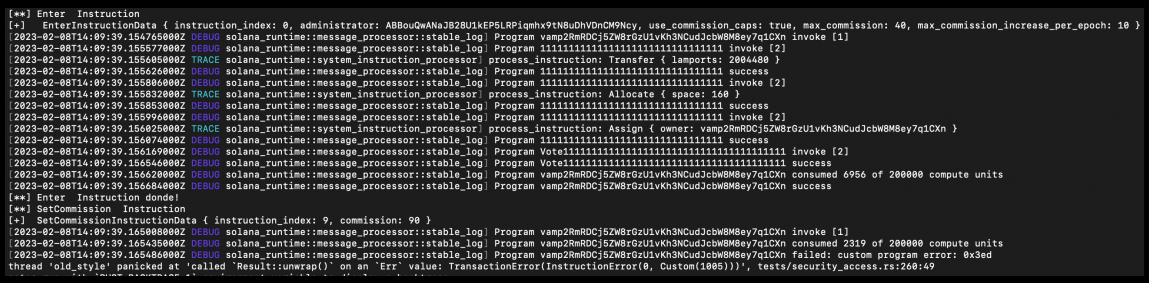
4.2 IMPROPERLY SET COMMISSION

Description:

If commission caps are enabled for a vote account, then the rewards' authority is restricted by those caps. This authority cannot set the commission to a new value higher than the max_commission value set with the Enter instruction. This restriction was tested to confirm no vulnerabilities were introduced.

Results:

No vulnerabilities were identified.



The error got in hexadecimal corresponds to the following one:

Listing 3: program/entrypoint.c (Line 247)

```
245 // Attempt to Enter for a vote account whose commission is already
 ↳  larger than the enforced maximum commission; or
246 // attmept to set commission to a value larger than the enforced
 ↳ maximum commission
247 Error_CommissionTooLarge                    = 1005,
```

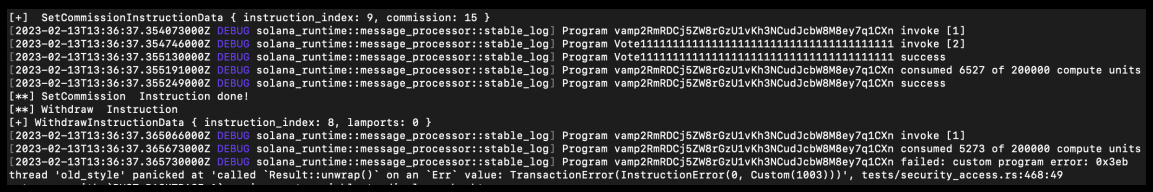
4.3 INCORRECT NUMBER OF ACCOUNT PROVIDED

Description:

All instructions require a specific number of accounts for each that must be provided for proper and safe functionality. A test has been carried out to ensure that the number of accounts is checked correctly and, if not correct, makes the transaction to fail.

Results:

No vulnerabilities were identified.



The error got in hexadecimal corresponds to the following one:

Listing 4: program/entrypoint.c (Line 242)

```
241 // The number of accounts that were suppled in the instruction was
 ↳  incorrect for the instruction type
242 Error_IncorrectNumberOfAccounts             = 1003,
```

4.4 SETTING AN INVALID LEAVE EPOCH

Description:

When commission caps are enabled, then the vote account cannot be removed from management by the Vote Account Manager program until a **leave epoch** authority to an epoch at least 2 beyond the current epoch, and once set. This restriction was tested to confirm it is safe, and no vulnerabilities were introduced.

Results:

No vulnerabilities were identified.

```
[+] SetLeaveEpochInstructionData { instruction_index: 1, leave_epoch: 1 }
SetLeaveEpochInstructionData { instruction_index: 1, leave_epoch: 1 }
[2023-02-13T16:10:38.692082000Z DEBUG solana_runtime::message_processor::stable_log] Program vamp2RmRDCj5ZW8rGzU1vKh3NCudJcbW8M8ey7q1CXn invoke [1]
[2023-02-13T16:10:38.692374000Z DEBUG solana_runtime::message_processor::stable_log] Program vamp2RmRDCj5ZW8rGzU1vKh3NCudJcbW8M8ey7q1CXn consumed 3803 of 200000 compute units
[2023-02-13T16:10:38.692408000Z DEBUG solana_runtime::message_processor::stable_log] Program vamp2RmRDCj5ZW8rGzU1vKh3NCudJcbW8M8ey7q1CXn failed: custom program error: 0x3f1
thread 'old_style' panicked at 'called `Result::unwrap()` on an `Err` value: TransactionError(InstructionError(0, Custom(1009)))', tests/security_access.rs:332:49
note: run with `RUST_BACKTRACE=1` environment variable to display a backtrace
```

The error got in hexadecimal corresponds to the following one:

Listing 5: program/entrypoint.c (Line 259)

```
258 // Attempt to set a leave epoch that is not at least one full
 ↳ epoch away
259 Error_InvalidLeaveEpoch                    = 1009,
260
```

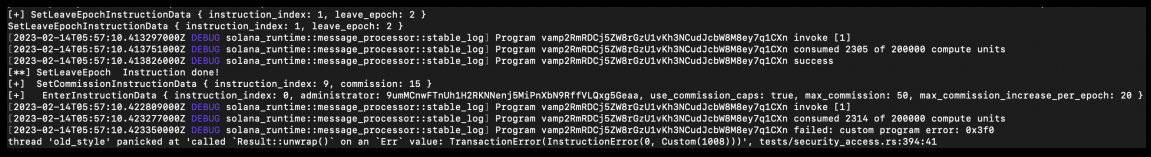
4.5 SETTING COMMISSION AFTER SETTING LEAVE EPOCH

Description:

When commission caps are enabled, then the vote account cannot be removed from management by the Vote Account Manager program until a **leave epoch** has been set at least 2 beyond the current epoch. Once set, it prevents any commission changes on the vote account from ever occurring again. This restriction was tested to confirm it is safe, and no vulnerabilities were introduced.

Results:

No vulnerabilities were identified.



The error got in hexadecimal corresponds to the following one:

Listing 6: program/entrypoint.c (Line 256)

```
255 // Attempt to set the leave epoch or commission when leave epoch
 ↳ was already set
256 Error_LeaveEpochAlreadySet                  = 1008,
```

4.6 LEAVE BEFORE THE CORRESPONDING EPOCH

Description:

When commission caps are enabled, then the vote account cannot be removed from management by the Vote Account Manager program until a **leave epoch** has been set at least 2 beyond the current epoch.
This restriction was tested to confirm it is safe, and no vulnerabilities were introduced.

Results:

No vulnerabilities were identified.

```
[+] SetLeaveEpochInstructionData { instruction_index: 1, leave_epoch: 2 }
SetLeaveEpochInstructionData { instruction_index: 1, leave_epoch: 2 }
[2023-02-13T16:15:09.379617000Z DEBUG solana_runtime::message_processor::stable_log] Program vamp2RmRDCj5ZW8rGzU1vKh3NCudJcbW8M8ey7q1CXn invoke [1]
[2023-02-13T16:15:09.379860000Z DEBUG solana_runtime::message_processor::stable_log] Program vamp2RmRDCj5ZW8rGzU1vKh3NCudJcbW8M8ey7q1CXn consumed 2305 of 200000 compute units
[2023-02-13T16:15:09.379902000Z DEBUG solana_runtime::message_processor::stable_log] Program vamp2RmRDCj5ZW8rGzU1vKh3NCudJcbW8M8ey7q1CXn success
[**] SetLeaveEpoch  Instruction done!
[+]  SetCommissionInstructionData { instruction_index: 9, commission: 15 }
[**] Withdraw  Instruction
[+] WithdrawInstructionData { instruction_index: 8, lamports: 0 }
[2023-02-13T16:15:09.385660000Z DEBUG solana_runtime::message_processor::stable_log] Program vamp2RmRDCj5ZW8rGzU1vKh3NCudJcbW8M8ey7q1CXn invoke [1]
[2023-02-13T16:15:09.386028000Z DEBUG solana_runtime::message_processor::stable_log] Program Vote111111111111111111111111111111111111111 invoke [2]
[2023-02-13T16:15:09.386153000Z DEBUG solana_runtime::message_processor::stable_log] Program Vote111111111111111111111111111111111111111 success
[2023-02-13T16:15:09.386201000Z DEBUG solana_runtime::message_processor::stable_log] Program vamp2RmRDCj5ZW8rGzU1vKh3NCudJcbW8M8ey7q1CXn consumed 3576 of 200000 compute units
[2023-02-13T16:15:09.386241000Z DEBUG solana_runtime::message_processor::stable_log] Program vamp2RmRDCj5ZW8rGzU1vKh3NCudJcbW8M8ey7q1CXn success
[**] Withdraw  Instruction donde!
[+] Leave  Instruction
[2023-02-13T16:15:09.393057000Z DEBUG solana_runtime::message_processor::stable_log] Program vamp2RmRDCj5ZW8rGzU1vKh3NCudJcbW8M8ey7q1CXn invoke [1]
[2023-02-13T16:15:09.393453000Z DEBUG solana_runtime::message_processor::stable_log] Program vamp2RmRDCj5ZW8rGzU1vKh3NCudJcbW8M8ey7q1CXn consumed 2657 of 200000 compute units
[2023-02-13T16:15:09.393493000Z DEBUG solana_runtime::message_processor::stable_log] Program vamp2RmRDCj5ZW8rGzU1vKh3NCudJcbW8M8ey7q1CXn failed: custom program error: 0x3f3
thread 'old_style' panicked at 'called `Result::unwrap()` on an `Err` value: TransactionError(InstructionError(0, Custom(1011)))', tests/security_access.rs:541:49
```

The error got in hexadecimal corresponds to the following one:

Listing 7: program/entrypoint.c (Line 265)

```
264 // Attempt to leave before the leave epoch
265 Error_CannotLeaveYet                          = 1011,
```

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 Semgrep, it is a fast, open-source, static analysis engine for finding bugs, detecting vulnerabilities in third-party dependencies, and enforcing code standards. Semgrep analyzes code locally in the computer or build environment.

Results:

```
Scanning 1 file with 35 c rules.


  Results

Findings:

  /Users/isabelburruezo/halborn/projects/Solana/voteAccount/vote-account-manager-master/program/entrypoint.c
     semgrep-rules-main.c.raptor-interesting-api-calls
        Locate all calls to interesting and potentially insecure API functions (candidate points).
        The auditor can backtrace from these candidate points to find pathways allowing access from
        untrusted input.

        538¦ void *memcpy(void *dst, const void *src, int len)
          ⋮¦----------------------------------------
        540¦ (void) sol_memcpy(dst, src, len);
          ⋮¦----------------------------------------
     semgrep-rules-main.c.raptor-typos
        The programmer accidentally uses the wrong operator, which changes the application logic in
        security-relevant ways. These types of errors are generally the result of a typo. This rule
        also covers some other common typo patterns.

        777¦ if (result || // sol_get_rent_sysvar failed, so u and exp are bogus
        778¦     (u & 0x8000000000000000ul) || // negative exemption_threshold
        779¦     ((exp == 0) || (exp == 0x7FF))) { // subnormal values
        780¦     // Unsupported and basically nonsensical rent exemption threshold.  Just use some hopefully sane default based
        781¦     // on historical values that were true for 2021/2022: lamports_per_byte_year = 3480, exemption_threshold = 2
        782¦     // years
        783¦     return (account_size + 128) * 3480 * 2;
        784¦ }


  Scan Summary

Ran 36 rules on 1 file: 3 findings.
```

THANK YOU FOR CHOOSING

// HALBORN