Audit Report

Audit Summary

Methodology

The codebase was reviewed manually by carefully examining every line of code to understand the functionality and identify potential vulnerabilities. During the review, it was observed that the code was well-written and maintained a high standard of quality. No major vulnerabilities were detected. However, a memory safety concern was identified, which requires attention to ensure optimal performance and avoid potential stack memory overflow issues.

Findings and Recommendations

Issue: Memory Safety Concerns

Solana programs operate with a fixed stack size limit, which can be restrictive when dealing with large or dynamically sized data structures. If these limitations are not handled effectively, the program may face stack memory overflow issues, potentially leading to runtime errors. The following instances in the codebase were identified where improvements could be made:

File: `ini_main_state.rs`

- **Line 63**:

  ```
  pub main_state: Account<'info, MainState>, // MainState account
  ```

  Recommendation:
  Wrap the `Account` in a `Box` type to allocate the data on the heap instead of the stack. This approach will save stack space, ensuring better handling of large data structures.
  Updated Code:

  ```
  pub main_state: Box<Account<'info, MainState>>, // MainState account
  ```

File: `transfer_ownership.rs`

- **Line 45**:

  ```
  pub main_state: Account<'info, MainState>,
  ```

  Recommendation:
  Use `Box` to handle the `Account` efficiently and avoid stack size limitations.
  Updated Code:

  ```
  pub main_state: Box<Account<'info, MainState>>,
  ```

File: `update_main_state.rs`

- **Line 88**:

  ```
  pub main_state: Account<'info, MainState>, // MainState account with new values
  ```

  Recommendation:
  Wrap the `Account` in a `Box` type for the same reasons as above.
  Updated Code:

  ```
  pub main_state: Box<Account<'info, MainState>>, // MainState account with new values
  ```

Impact and Mitigation

By wrapping the `Account` type in a `Box`, the data is allocated on the heap instead of the stack, ensuring that the program can handle larger and more complex

structures. This change will enhance the program's robustness by preventing stack memory overflow while adhering to Solana's memory management requirements.

Conclusion

The codebase demonstrates strong security principles and good practices. Addressing the identified memory safety concerns will further enhance the program's reliability and performance. It is recommend to implement the above changes to verify the updates.