FYEO

Banger

Security Review Update: March 25, 2025

Reviewer: balthasar@gofyeo.com

Banger Security Review Update

New security issues, 2

After the development team implemented the latest updates, FYEO conducted a review of the modifications. The primary goal of this evaluation was to ensure the continued robustness of the program's security features, safeguarding user funds and maintaining the overall integrity of the program.

General Updates:

This recent update adds two new instructions to the Banger program that allow modification of the URIs of assets and collections on Metaplex. The `UpdateToken` instruction is designed to update the metaplex token data. It ensures that only an account with administrative privileges, identified by a specific `AUTHORITY`, can execute these updates.

The `WithdrawLiquidity` instruction calculates and mints remaining token supplies to both the DEX admin and the author vault, ensuring that the correct amounts are allocated based on predefined constants (`TOTAL_SUPPLY`, `DEX_LIQUIDITY_SUPPLY_THRESHOLD`, and `AUTHOR_ALLOCATION`). After minting tokens, the instruction revokes the mint authority of the token to prevent further minting.

Other modifications have been done such as a launcher account paying for account creations. These do not affect the security of the program.

While the new instructions are securely implemented, no new tests have been added. This should be addressed, as tests are essential to maintaining a robust and reliable codebase by catching regressions early, ensuring consistent functionality, and providing confidence during future enhancements.

Specific Security Changes:

The `UpdateToken` and `WithdrawLiquidity` instructions require an authorized admin account. This setup ensures that only accounts with specific `ADMIN_PUBKEY` can make changes, securely limiting updates to authorized users only.

Init if needed ATAs could have other authorities

Finding ID: FYEO-BANGER-01
Severity: Low
Status: Remediated

Description

The `WithdrawLiquidity` handler has two `init_if_needed` associated token accounts (`author_vault_ata`, `dex_admin_token_account`) while this is an admin function, it could be possible to pass in some ATA (associated token account) not owned by the expected users - since the program is using the Tokenkeg program, the authority might be different to that used to derive the ATA's address.

Proof of Issue

File name: programs/banger-program/src/instructions/withdraw_liquidity.rs
Line number: 55

```
#[account(
    init_if_needed,
    payer = admin,
    associated_token::mint = mint,
    associated_token::authority = author_vault,
)]
pub author_vault_ata: Box<Account<'info, TokenAccount>>,

...

#[account(
    init_if_needed,
    payer = admin,
    associated_token::mint = mint,
    associated_token::authority = dex_admin,
)]
pub dex_admin_token_account: Box<Account<'info, TokenAccount>>,
```

Severity and Impact Summary

If the admin is not careful, they might sign a transaction with an ATA that is not owned by the assumed authority.

Recommendation

Make sure to check that the authority is correct. Also consider checking for delegate authorities being set on these accounts.

Account is not the ATA but documented to be

Finding ID: FYEO-BANGER-02
Severity: Informational
Status: Remediated

Description

Claim author rewards documents that `author_vault_ata` is the associated token account (ATA) - which it is not - it accepts any token account owned by the expected authority.

Proof of Issue

File name: programs/banger-program/src/instructions/claim_author_rewards.rs
Line number: 50

```
// Associated token account of the author vault, optional to claim from the
author vault
#[account(
    mut,
    token::mint = mint,
    token::authority = author_vault,
)]
pub author_vault_ata: Option<Box<Account<'info, TokenAccount>>>,
```

Severity and Impact Summary

This accepts claims to any token account of the correct authority.

Recommendation

Make sure this is only accepting the correct account.

Commit Hash Reference:

For transparency and reference, the security review was conducted on the specific commit hash for the Banger repository. The commit hash for the reviewed versions is as follows: 62d975fd5b656f89faac4918460e8f8f1d6668f2

Remediations were completed with the commit 8f068f0b78dfb1d041ca310249b46a5368b3008b.

Conclusion:

In conclusion, the security aspects of the Banger program remain robust and unaffected by the recent updates. Users can confidently interact with the protocol, assured that their funds are well-protected. The commitment to security exhibited by the development team is commendable, and we appreciate the ongoing efforts to prioritize the safeguarding of user assets.