ENIGMA DARK

Securing the Shadows

Security Review

Asterix Labs

April, 2024

Summary

Asterix

Asterix is a protocol focused on unlocking new possibilities of digital ownership, defining the standards for the future by leveraging innovative standards such as the DN404 and ERC6551.

The contracts of Asterix, hosted at repo, were reviewed over 7 days, between April 1st and April 7th, 2024. The protocol was reviewed at commit a0a6156e7d08636e11a9047a8ed4cd9daec27f49.

The scope of the security review consisted of the following contracts at the specific commit:

```
src/
├── vault
|   ├── Vault.sol
|   └── VaultManager.sol
├── vesting
|   ├── VestingVault.sol
|   ├── VestingVaultProxy.sol
|   └── VestingVaultManager.sol
└── staking
    ├── StakingVault.sol
    ├── StakingVaultProxy.sol
    └── StakingVaultManager.sol
```

Disclaimer

This security review does not guarantee against a hack. It is a snapshot in time according to the specific commit. Enigma Dark makes no warranties regarding the security of the code and does not warrant that the code is free from defects. By deploying or using the code, Asterix and users of the contracts agree to use the code at their own risk. Any modifications to the code will require a new security review.

Critical Findings

None.

High Findings

None.

Medium Findings

None.

Low Findings

[L-01] - `VaultManager` function `tokenURI` returns early if used within the same contract

Severity: Low risk

Context: VaultManager.sol#L103-L129

Description: Instead of the conventional return statement at the end of the `tokenURI` execution, the function uses `LibString.directReturn()` to save gas. This alternative method doesn't copy data to memory; rather, it returns data to the external context and halts execution. Consequently, this renders the function inaccessible for internal calls, limiting its utilization under the `public` visibility modifier. Within the current state of the codebase, no impact has been observed. However, if any future contracts inheriting from `VaultManager` call `tokenURI` before additional code, the execution would prematurely halt.

Recommendation: Consider either setting the function visibility external or using normal return at the expense of using more gas.

Vectorized: Noted. Unfortunately, Solidity throws an error if we try to change the visibility of the override from public to external.

[L-02] - Centralization risk

Severity: Low risk

Context: VestingVaultManager.sol#L153-L197

Description: Within the `VestingVaultManager` contract, admin operations happen in two steps: first, a function is used to set a value top set a value and a function to lock the value on storage, making it immutable. This raises concerns that, until the locking process is initiated, a malicious administrator could freely alter any value.

Recommendation: Consider locking the values in the contract storage in the same transaction that sets them.

Vectorized: Noted.

Gas Optimization Findings

[G-01] - `VestingVault` does not follow early revert pattern

Severity: Gas

Context: VaultManager.sol#L59-L80

Description: The functions `_afterExecute` & `_afterExecuteBatch` are designed to revert the execution only after the main `ERC6551.execute` & `ERC6551.executeBatch` logic has been executed and gas has been spent. This approach does not adhere to the early revert pattern, which suggests performing checks before the main part of the function. By doing so, if a revert occurs, the user does not expend extra gas on an unnecessary execution. This issue is particularly noticeable in the `executeBatch` function.

Recommendation: Consider changing the after hooks for before hooks in order to revert early.

Vectorized: The hooks are designed to be called at the end of the function, as we initially wanted to have the flexbility to add in some invariant checks at the end. But in the end, we did not use it.
Since our test code has already been structured to expect the hooks at the end, and the VestingVault is for internal use, we are ok with leaving it as it is.

Informational Findings

[I-01] - Lack of event emission after sensitive changes and interactions

Severity: Informational

Context: StakingVaultManager.sol#L7, VaultManager.sol#L11, VestingVaultManager.sol

Description: The `lockVaultProxy`, `setVaultProxy` in `StakingVaultManager`, `_initializeVaultManager`, `setBaseURI`, `setTokenURIRenderer`, `setNameAndSymbol`, `createVaultFor`, `depositAsterixERC20For`, `depositAsterixMirrorERC721For`, `withdrawAsterixERC20To` and `withdrawAsterixMirrorERC721To` in `VaultManager`, and `lockVaultProxy`, `setVaultProxy`, `lockCliffDuration`, `setCliffDuration`, `lockVestingDuration`, `setVestingDuration`, `lockEarlyVestingDuration`, `setEarlyVestingDuration`, `startCliff` and `startEarlyVesting` in `VestingVaultManager` lack event emission, which might affect third-party services and users that track state changes in the contract in an off-chain manner.

Recommendation: Consider emitting an event whenever significant parameters of each contract are modified, or whenever relevant interactions are performed, enabling seamless communication with clients regarding state changes and facilitating application subscriptions to these updates.

Vectorized: Currently, our backend and frontend do not listen for events for admin functions.
We are ok with not adding the events.

[I-02] - Functions specific for testnet should be removed

Severity: Informational

Context: VestingVaultManager.sol#L221, VestingVaultManager.sol#L236, VestingVaultManager.sol#L247

Description: The `directResetVestingState` and `directSetInitialLockedAsterixERC20Amount` functions, together with the `_requireOnlyTestnet` are designed to only be triggered in the Sepolia/Holesky testnets. Although this does not cause any security issue as such functions can't be called in other chains, the contract's size can be greatly reduced by directly removing such functions instead of deploying the contract with them.

Recommendation: Consider removing the testnet functions from the `VestingVaultManager` contract. If intended for testing, create and additional `VestingVaultManagerTestnet` contract that inherits from `VestingVaultManager` and incorporates the testnet-only mentioned functions.

Vectorized: Noted.

[I-03] - Unused internal function

Severity: Informational

Context: VaultManager.sol#L319-L322

Description: `_setSkipNFT` is an internal function not used anywhere in the codebase.

Recommendation: Consider removing the function in the case it is not planned to be used in any of the contracts inheriting `VaultManager`.

Vectorized: Noted. We will leave it in there, as it is used in tests.