STATE MIND

Arrakis v2 manager templates

17-05-2023 - 31-08-2023

Table of contents

1. Project Brief

Title	Description
Client	Arrakis
Project name	Arrakis v2 manager templates
Timeline	17-05-2023 – 31-08-2023
Initial commit	1d507a2beaa9c0e785bac7dd943c77964fedaef3
Final commit	273a0b4f263f7737e1bbae647cbc4840fd90d5e9

Project Scope

The audit covered the following files:

ChainLinkOracle.sol

UniswapV3PoolOracle.sol

SimpleManager.sol

2. Finding Severity breakdown

All vulnerabilities discovered during the audit are classified based on their potential severity and have the following classification:

Severity	Description
Critical	Bugs leading to assets theft, fund access locking, or any other loss of funds to be transferred to any party.
High	Bugs that can trigger a contract failure. Further recovery is possible only by manual modification of the contract state or replacement.
Medium	Bugs that can break the intended contract logic or expose it to DoS attacks, but do not cause direct loss of funds.
Informational	Bugs that do not have a significant immediate impact and could be easily fixed.

Based on the feedback received from the Customer regarding the list of findings discovered by the Contractor, they are assigned the following statuses:

Status	Description
Fixed	Recommended fixes have been made to the project code and no longer affect its security.
Acknowledged	The Customer is aware of the finding. Recommendations for the finding are planned to be resolved in the future.

3. Summary of findings

Severity	# of Findings
Critical	0 (0 fixed, 0 acknowledged)
High	0 (0 fixed, 0 acknowledged)
Medium	3 (3 fixed, 0 acknowledged)
Informational	8 (5 fixed, 3 acknowledged)
Total	11 (8 fixed, 3 acknowledged)

4. Conclusion

During the audit of Arrakis v2 manager templates codebase, 11 issues were found in total:

- 3 medium severity issues (3 fixed)
- 8 informational severity issues (5 fixed, 3 acknowledged)

The final reviewed commit is 273a0b4f263f7737e1bbae647cbc4840fd90d5e9.

Contracts are deployed on ethereum, arbitrum, base, bsc, optimism, polygon networks under the same addresses.

Deployment

File name	Contract deployed on mainnet
DefaultProxyAdmin	0x89dc63264dAB74a4350D7F44bD62eec3b22c9ca0
SimpleManager (proxy)	0x3522Df5C13a40dfaA9CEc17E12F5fbD29dc811E1
SimpleManager (implementation)	0x64AB6C28423Bd60611199a01c6720a0576D9a9fa

5. Findings report

MEDIUM-01	Revert for sqrtPriceX96 >= 2^128	Fixed at f0c861

Description

Within the **SimpleManager** contract, the **rebalance** function includes a price calculation that triggers a revert if **sqrtPriceX96** is greater than or equal to 2^128. The calculation involves squaring **sqrtPriceX96**, and if the result is larger than or equal to 2^256, an overflow occurs, leading to the revert.
This issue also affects the following functions:

- Twap.sol#L45
- Twap.sol#L60

Recommendation

It is recommended to fix the calculation by referring to **this function**.

MEDIUM-02	Improper Chainlink oracle handling	Fixed at 74717e

Description

The **ChainLinkOracle** contract exhibits issues in the handling of functions such as **getPrice0** and **getPrice1** due to improper handling of the **latestRoundData()** function.

1. There is no check to ensure that the **updatedAt** variable is within an acceptable time range. This omission may result in the usage of outdated and unreliable data.
2. The implementation does not account for Layer 2 solutions that require an additional Sequencer contract to provide data. Further details can be found here.
3. There is no try-catch block to handle cases where the price feed reverts instead of responding, resulting in a revert of the entire transaction.

Recommendation

To address these issues, it is recommended to make the following improvements:

1. Implement a check to ensure that the **updatedAt** variable is within an acceptable time range, discarding outdated data.
2. Consider adapting the solution to accommodate Layer 2 scenarios with the use of a Sequencer contract.
3. Implement a try-catch block to handle cases where the price feed reverts, allowing for graceful handling of such scenarios. Additionally, consider using a secondary oracle, such as UniswapV3's, to provide a fallback option in case of a revert from the primary oracle.

MEDIUM-03	Rounding error in ChainLinkOracle	Fixed at 74717e

Description

In the getPrice0 method of ChainLinkOracle contract, there is a rounding error in the price calculation:

```
FullMath.mulDiv(
    FullMath.mulDiv(
        10 ** priceFeedDecimals,
        10 ** priceFeedDecimals,
        SafeCast.toUint256(price)
    ),
    10 ** token1Decimals,
    10 ** priceFeedDecimals
);
```

Consider 2 cases when **priceFeedDecimals < token1Decimals**:

More realistic less significant:

1. **priceFeedDecimals = 5**
2. **token1Decimals = 6**
3. **price = 846813**
4. the resulting price is **118080**
5. actual price should be **118089**

Less realistic more significant:

1. **priceFeedDecimals = 2**
2. **token1Decimals = 6**
3. **price = 249377**
4. the resulting price is **0**
5. actual price should be **400**

The error occurs cause rounding is performed before multiplication by **10 ** token1Decimals**

Also, getPrice1 method has the same error.

Recommendation

We recommend performing multiplication before division like

```
FullMath.mulDiv(
    FullMath.mulDiv(
        10 ** (2 * priceFeedDecimals),
        10 ** token1Decimals,
        SafeCast.toUint256(price)
    ),
    1
    10 ** priceFeedDecimals
);
```

INFORMATIONAL-01	Gas optimizations	Fixed at 74717e

Description

1. Use **++i** instead of **i++**
    - SimpleManager.sol#L143
    - SimpleManager.sol#L178
    - SimpleManager.sol#L213
    - SimpleManager.sol#L223
    - SimpleManager.sol#L236
    - SimpleManager.sol#L251
    - SimpleManager.sol#L348

- SimpleManager.sol#L143
- SimpleManager.sol#L213
- SimpleManager.sol#L223
- SimpleManager.sol#L236
- SimpleManager.sol#L251

3. Delete a "dead" code
   - SimpleManager.sol#L357-L370
4. Remove a useless check
   - SimpleManager.sol#L208
   - SimpleManager.sol#L269
   - SimpleManager.sol#L347

Recommendation

It is recommended to optimize the following areas.

INFORMATIONAL-02	Zero address check	Fixed at 74717e

Description

Several functions in the codebase lack zero address checks for their input parameters. The affected functions are:

- SimpleManager.sol#L73
- SimpleManager.sol#L77
- ChainLinkOracle.sol#L23
- UniswapV3PoolOracle.sol#L16

Recommendation

It is recommended to include these zero address checks.

INFORMATIONAL-03	No checks if vault is in vaults	Fixed at 74717e

Description

1. In the function **SimpleManager.rebalance()**, an operator can burn liquidity from a vault in which **SimpleManager** is the manager, but it **initManagement()** was not called yet.
2. In the function **SimpleManager.withdrawAndCollectFees()**, the owner can withdraw manager balance from a vault in which **SimpleManager** is the manager, but it **initManagement()** was not called yet.

Recommendation

We recommend adding checks if vault is in **vaults**.

INFORMATIONAL-04	Unused function in SimpleManager	Fixed at 74717e

Description

Method _includesAddress is unused.

Recommendation

We recommend removing this method.

INFORMATIONAL-05	Useless import	Fixed at 74717e

Description

ERC20 import in **SimpleManager.sol** contract isn't used. **Range** import is not needed, cause it's included in **Rebalance** struct.

Recommendation

We recommend removing these import.

INFORMATIONAL-06	No way to remove vaults and modify vault info	Acknowledged

Description

The function **SimpleManager.initManagement()** adds a vault to **vaults** and sets its info. However, there is no way to undo this action or modify vault info after **initManagement()**. We recommend to add a function to remove a vault from **vaults** that is callable by the **SimpleManager** owner. One needs to consider the possibility of vault owners frontrunning the **rebalance()** function. This way, modifying vault info would require the **SimpleManager** owner to remove the vault and the vault owner to add the vault again with different parameters.

Recommendation

We recommend to add a function to remove a vault from **vaults**.

INFORMATIONAL-07	Max deviation is not bounded	Acknowledged

Description

In the function **SimpleManager.initManagement()**, the parameter **params.maxDeviation** is not bounded from above. If the vault owner sets the wrong parameter, an operator can sandwich the rebalance by swapping outside of the rebalance.

Recommendation

We recommend to add an upper limit for **params.maxDeviation**.

INFORMATIONAL-08	Code duplicating	Acknowledged

Description

In **rebalance()** function there're two calls to oracle for getting the price of **token0**. In **if** condition and in **_checkMinReturn** internal call. So, it can be avoided by making **oraclePrice** argument instead of sending **vaultInfo.oracle** to this internal function and moving to get price outside **if** condition, cause, as was mentioned before it should always be checked.

Recommendation

We recommend fixing this double call, it will be easily if **simpleManager** will always check **poolPrice** and **oraclePrice** deviation.

STATE
MIND