**MDC.sol**

# **High severity issues**

* **Get token price**

In the function tokenPrice, we are using the DEX contract to calculate the token price and use this price in the function upgradeAmount. The attacker can use the flash loan and manipulate the price of the token.



* **Recommendation**

Use oracle service to get the price.

# **Medium severity issues**

* **MINTER\_ROLE / Centralization**

In the function mint, owner or role \_allowMint, can mint tokens and distribute those tokens without obtaining the consensus of the community, this could be a centralization risk.

In the function setAllowint, the owner role can set any address as role \_ allowMint. allowMint, can mint tokens and distribute those tokens without obtaining the consensus of the community, this could be a centralization risk.



Text

Description automatically generated

The owner role has access to the below functions and can make changes to contract storage variables. These variables are related to user actions.

setV2Pair, unsetV2Pair, setAllowMint, setSwaptime, setExcluded, setMaxDao

* **Recommendation**

The risk describes the current project design and potentially makes iterations to improve the security operation and level of decentralization, which in most cases cannot be resolved entirely at the present stage. We advise the client to carefully manage the privileged account's private key to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol be improved via a decentralized mechanism or smart-contract-based accounts with enhanced security practices, e.g., multi-signature wallets

# **Low severity issues**

* **Timestamp Dependency**

There is a Timestamp Dependence problem in \_transfer. The timestamp of a block, accessed by now or block.timestamp can be manipulated by a miner.



* **Recommendation**

Use oracle service to get the time.

* **Missing Emit Events**

**There should always be events emitted in the sensitive functions.**

* **function mint(address \_uid, uint256 \_tokens) external returns (bool);**
* **function transfer(address recipient, uint256 amount);**
* **…**
* **Recommendation**

**It is recommended emitting events for sensitive functions.**

**MaxDAO.sol**

# **High severity issues**

* **Get token price**

In the function tokenPrice, we are using the DEX contract to calculate the token price and use this price in the function related to deposit in platform. The attacker can use the flash loan and manipulate the price of the token.

Text

Description automatically generated

* **Recommendation**

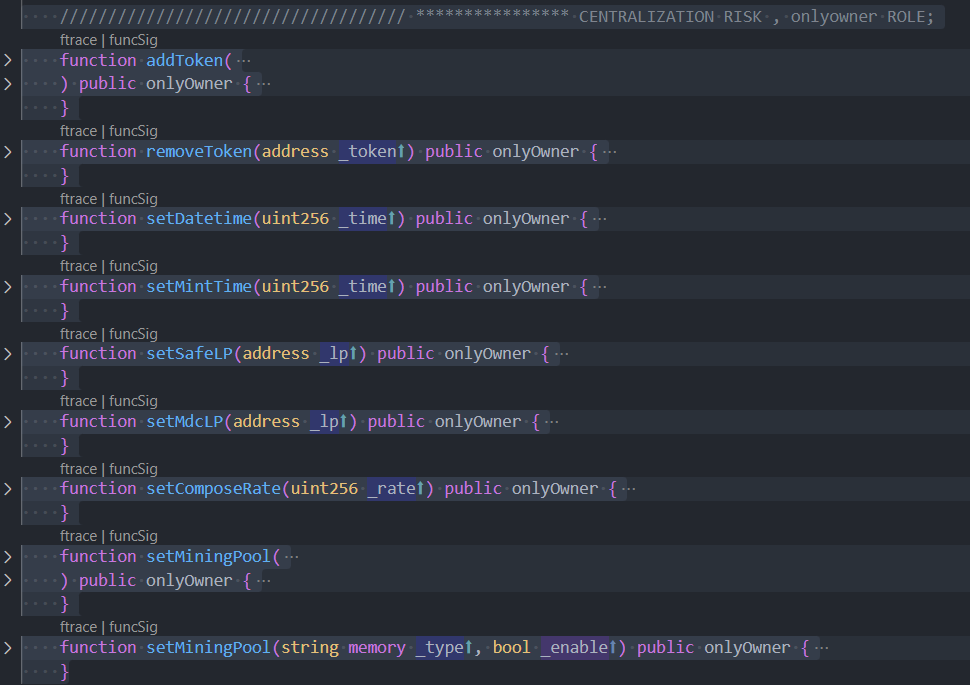
Use oracle service to get the price.

# **Medium severity issues**

* **OWNER\_ROLE / Centralization**

In the functions listed below, the only owner role has access to the functions and can make changes to contract storage variables. These variables are related to user actions.

For example, in one scenario, the owner can add a token contract containing a Malicious code or the owner can remove the token from the contract without the users' permission.



* **Recommendation**

The risk describes the current project design and potentially makes iterations to improve the security operation and level of decentralization, which in most cases cannot be resolved entirely at the present stage. We advise the client to carefully manage the privileged account's private key to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol be improved via a decentralized mechanism or smart-contract-based accounts with enhanced security practices, e.g., multi-signature wallets.

# **Low severity issues**

* **Timestamp Dependency**

There is a Timestamp Dependence problem in withdrawToken. The timestamp of a block, accessed by now or block.timestamp can be manipulated by a miner.

Text

Description automatically generated

* **Recommendation**

Use oracle service to get the time.

* **Unchecked returned value**

In the functions below, the contract is using the transfer function without checking returned value.



* **Recommendation**

We recommend checking the returned value or using safeERC20library.

* **Missing Emit Events**

**There should always be events emitted in the sensitive functions.**

* **function depositAlone at line 745.**
* **function transfer at line 720.**
* **function depositCompose at line 768.**
* **…**
* **Recommendation**

**It is recommended emitting events for sensitive functions.**