

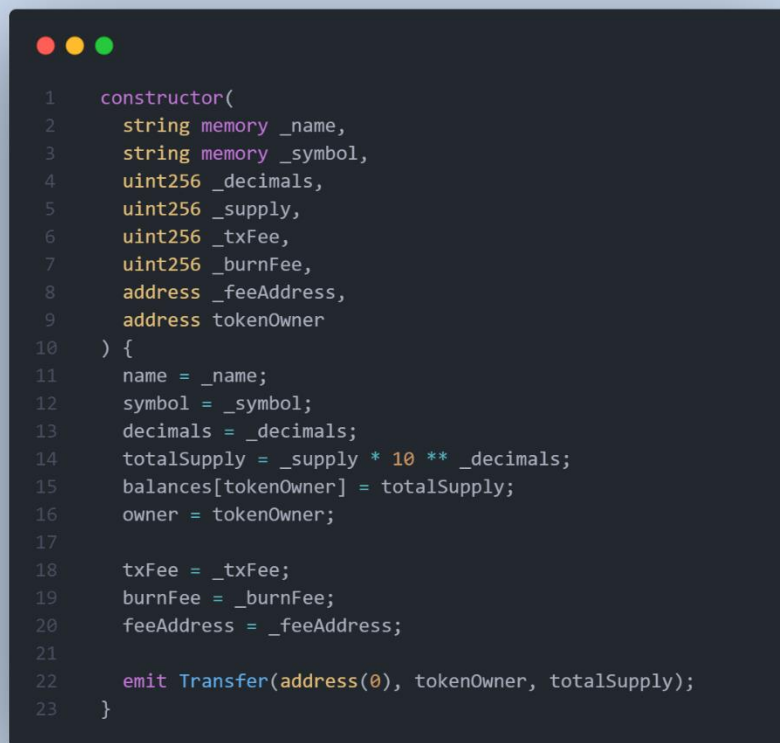
Original Audit :

<https://www.certik.com/projects/equality>

High severity issues

- **CENTRALIZATION RISKS**

In the constructor function, all supply tokens are transferred to the deployer or owner of the contract.



```
1  constructor(
2      string memory _name,
3      string memory _symbol,
4      uint256 _decimals,
5      uint256 _supply,
6      uint256 _txFee,
7      uint256 _burnFee,
8      address _feeAddress,
9      address tokenOwner
10 ) {
11     name = _name;
12     symbol = _symbol;
13     decimals = _decimals;
14     totalSupply = _supply * 10 ** _decimals;
15     balances[tokenOwner] = totalSupply;
16     owner = tokenOwner;
17
18     txFee = _txFee;
19     burnFee = _burnFee;
20     feeAddress = _feeAddress;
21
22     emit Transfer(address(0), tokenOwner, totalSupply);
23 }
```

Remediation

We recommend the team be transparent regarding the initial token distribution process, and the team shall make enough efforts to restrict the access of the private key.

Medium severity issues

- **ZERO ADDRESS / VALUE VALIDATION CHECK**

In the constructor functions shown below, the deployer is passing values and addresses as arguments and set them as important storage states of the contract. These states are related to user transactions and it's important to check them before make store them in the contract.

```
1  constructor(
2      string memory _name,
3      string memory _symbol,
4      uint256 _decimals,
5      uint256 _supply,
6      uint256 _txFee,
7      uint256 _burnFee,
8      address _feeAddress,
9      address tokenOwner
10 ) {
11     name = _name;
12     symbol = _symbol;
13     decimals = _decimals;
14     totalSupply = _supply * 10 ** _decimals;
15     balances[tokenOwner] = totalSupply;
16     owner = tokenOwner;
17
18     txFee = _txFee;
19     burnFee = _burnFee;
20     feeAddress = _feeAddress;
21
22     emit Transfer(address(0), tokenOwner, totalSupply);
23 }
```

Remediation

We need to check the arguments in the constructor functions before making any changes to the contract. The address arguments should not be addressed 0 or uint256 arguments should not be 0 or more than the maximum allowed.

Low severity issues

- **IMPROPER USAGE OF PUBLIC AND EXTERNAL TYPE**

public functions that are never called by the contract could be declared as external. external functions are more efficient than public functions.

```
function burn(uint256 _value)
```

Remediation

Consider using the external attribute for public functions that are never called within the contract.