Equitas – Security Audit

1. SUMMARY

Equitas [EQT] is a deflationary charity token on Binance Smart Chain (BSC).

Audited Contract-Address on BSC mainnet:	0x180bDdD32C38632751bf40af484268e430BCe7D8
Website:	www.equitasinitiative.com
Whitepaper:	www.equitasinitiative.com/s/equitaswhitepaper.pdf
Twitter:	@officialequitas
Reddit:	@equitasinitiative
Telegram-Community:	@officialequitasinitiative

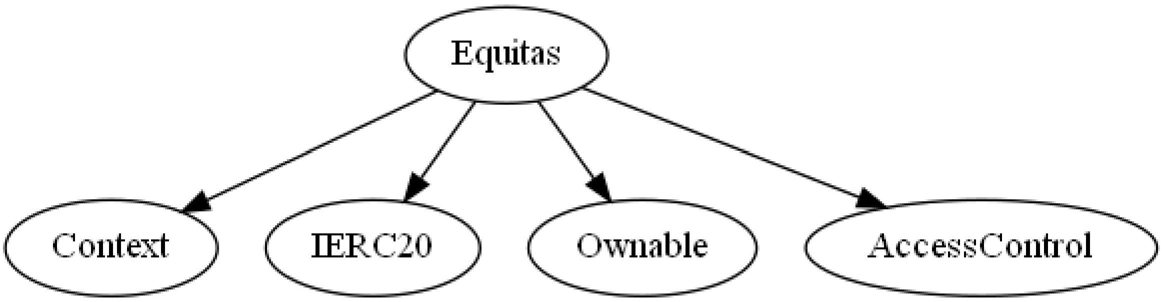
No major security issues identified.

July 8th, 2021

2. Overview of the Contract:

- Total supply of the token is set to one quadrillion.
- No mint or burn functions present. Circulating supply can be reduced by sending tokens to the 0x0..DEAD address, if desired.
- Liquidity locked for 6 months via Unicrypt.
- A role-based access model is used for executing the charity function following the OpenZeppelin best-practices.
- Ownership not renounced.
- There is a limit for transfers, but it is set to the total supply, thus this is merely informational.
- The contract uses SafeMath libraries to prevent overflows along with following the BEP20 standard.

3. Inheritence graph

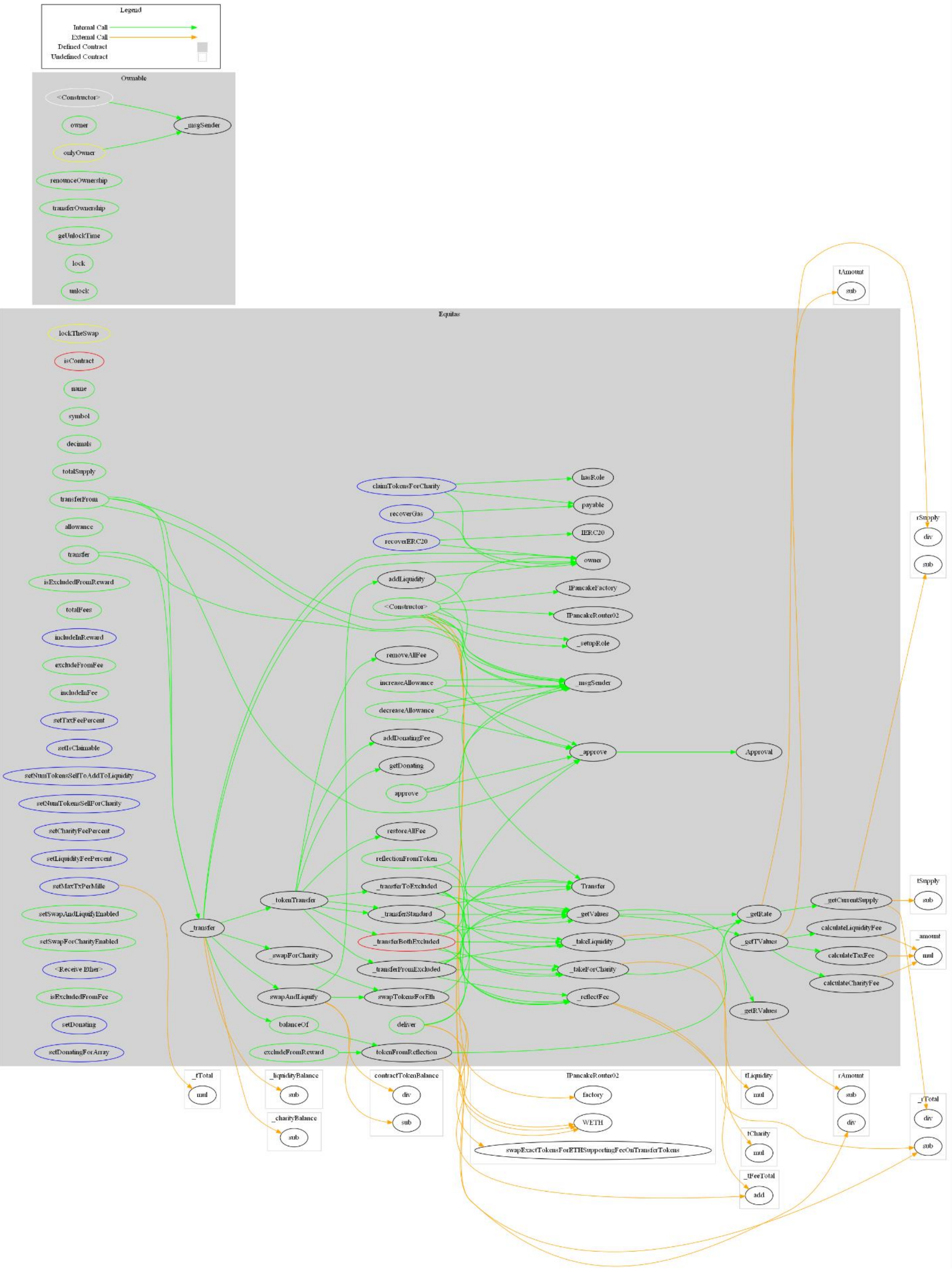


4. Contract-Description:

```
+  Equitas (Context, IERC20, Ownable, AccessControl)
    - [Pub] <Constructor> #
    - [Ext] claimTokensForCharity #
    - [Ext] recoverERC20 #
       - modifiers: onlyOwner
    - [Ext] recoverGas #
       - modifiers: onlyOwner
    - [Prv] isContract
    - [Pub] name
    - [Pub] symbol
    - [Pub] decimals
    - [Pub] totalSupply
    - [Pub] balanceOf
    - [Pub] transfer #
    - [Pub] allowance
    - [Pub] approve #
    - [Pub] transferFrom #
    - [Pub] increaseAllowance #
    - [Pub] decreaseAllowance #
    - [Pub] isExcludedFromReward
    - [Pub] totalFees
    - [Pub] deliver #
    - [Pub] reflectionFromToken
    - [Pub] tokenFromReflection
    - [Pub] excludeFromReward #
       - modifiers: onlyOwner
    - [Ext] includeInReward #
       - modifiers: onlyOwner
    - [Prv] _transferBothExcluded #
    - [Pub] excludeFromFee #
       - modifiers: onlyOwner
    - [Pub] includeInFee #
       - modifiers: onlyOwner
    - [Ext] setTaxFeePercent #
       - modifiers: onlyOwner
    - [Ext] setIsClaimable #
       - modifiers: onlyOwner
    - [Ext] setNumTokensSellToAddToLiquidity #
       - modifiers: onlyOwner
    - [Ext] setNumTokensSellForCharity #
       - modifiers: onlyOwner
    - [Ext] setCharityFeePercent #
       - modifiers: onlyOwner
    - [Ext] setLiquidityFeePercent #
       - modifiers: onlyOwner
    - [Ext] setMaxTxPerMille #
       - modifiers: onlyOwner
    - [Pub] setSwapAndLiquifyEnabled #
       - modifiers: onlyOwner
    - [Pub] setSwapForCharityEnabled #
       - modifiers: onlyOwner
    - [Ext] <Fallback> ($)
    - [Prv] _reflectFee #
    - [Prv] _getValues
    - [Prv] _getTValues
    - [Prv] _getRValues
    - [Prv] _getRate
    - [Prv] _getCurrentSupply
    - [Prv] _takeLiquidity #
    - [Prv] _takeForCharity #
    - [Prv] addDonatingFee #
    - [Prv] calculateTaxFee
    - [Prv] calculateLiquidityFee
    - [Prv] calculateCharityFee
    - [Prv] removeAllFee #
    - [Prv] restoreAllFee #
    - [Pub] isExcludedFromFee
    - [Prv] _approve #
    - [Prv] _transfer #
    - [Prv] swapAndLiquify #
       - modifiers: lockTheSwap
    - [Prv] _swapForCharity #
       - modifiers: lockTheSwap
    - [Prv] swapTokensForEth #
    - [Prv] addLiquidity #
    - [Prv] _tokenTransfer #
    - [Prv] _transferStandard #
    - [Prv] _transferToExcluded #
    - [Prv] _transferFromExcluded #
    - [Ext] setDonating #
       - modifiers: onlyOwner
    - [Ext] setDonatingForArray #
       - modifiers: onlyOwner
    - [Pub] getDonating


 ($) = payable function
 # = non-constant function
```

5. Contract flow graph



6. Technical overview

```
|  File Name  |  SHA-1 Hash  |
| Equitas.sol | 16c040d28dc51cd237a0f3dfa3256078ffab51df |


|  Contract  |  Type  |  Bases |
|  Function Name  |  Visibility  |  Mutability  |  Modifiers  |

| Equitas | Implementation | Context, IERC20, Ownable, AccessControl
| └ | <Constructor> | Public ❗ | 🛑  |NO❗ |
| └ | claimTokensForCharity | External ❗ | 🛑  |NO❗ |
| └ | recoverERC20 | External ❗ | 🛑  | onlyOwner |
| └ | recoverGas | External ❗ | 🛑  | onlyOwner |
| └ | isContract | Private 🔐 |   | |
| └ | name | Public ❗ |   |NO❗ |
| └ | symbol | Public ❗ |   |NO❗ |
| └ | decimals | Public ❗ |   |NO❗ |
| └ | totalSupply | Public ❗ |   |NO❗ |
| └ | balanceOf | Public ❗ |   |NO❗ |
| └ | transfer | Public ❗ | 🛑  |NO❗ |
| └ | allowance | Public ❗ |   |NO❗ |
| └ | approve | Public ❗ | 🛑  |NO❗ |
| └ | transferFrom | Public ❗ | 🛑  |NO❗ |
| └ | increaseAllowance | Public ❗ | 🛑  |NO❗ |
| └ | decreaseAllowance | Public ❗ | 🛑  |NO❗ |
| └ | isExcludedFromReward | Public ❗ |   |NO❗ |
| └ | totalFees | Public ❗ |   |NO❗ |
| └ | deliver | Public ❗ | 🛑  |NO❗ |
| └ | reflectionFromToken | Public ❗ |   |NO❗ |
| └ | tokenFromReflection | Public ❗ |   |NO❗ |
| └ | excludeFromReward | Public ❗ | 🛑  | onlyOwner |
| └ | includeInReward | External ❗ | 🛑  | onlyOwner |
| └ | _transferBothExcluded | Private 🔐 | 🛑  | |
| └ | excludeFromFee | Public ❗ | 🛑  | onlyOwner |
| └ | includeInFee | Public ❗ | 🛑  | onlyOwner |
| └ | setTaxFeePercent | External ❗ | 🛑  | onlyOwner |
| └ | setIsClaimable | External ❗ | 🛑  | onlyOwner |
| └ | setNumTokensSellToAddToLiquidity | External ❗ | 🛑  | onlyOwner |
| └ | setNumTokensSellForCharity | External ❗ | 🛑  | onlyOwner |
| └ | setCharityFeePercent | External ❗ | 🛑  | onlyOwner |
| └ | setLiquidityFeePercent | External ❗ | 🛑  | onlyOwner |
| └ | setMaxTxPerMille | External ❗ | 🛑  | onlyOwner |
| └ | setSwapAndLiquifyEnabled | Public ❗ | 🛑  | onlyOwner |
| └ | setSwapForCharityEnabled | Public ❗ | 🛑  | onlyOwner |
| └ | <Receive Ether> | External ❗ | 💵 |NO❗ |
| └ | _reflectFee | Private 🔐 | 🛑  | |
| └ | _getValues | Private 🔐 |   | |
| └ | _getTValues | Private 🔐 |   | |
| └ | _getRValues | Private 🔐 |   | |
| └ | _getRate | Private 🔐 |   | |
| └ | _getCurrentSupply | Private 🔐 |   | |
| └ | _takeLiquidity | Private 🔐 | 🛑  | |
| └ | _takeForCharity | Private 🔐 | 🛑  | |
| └ | addDonatingFee | Private 🔐 | 🛑  | |
| └ | calculateTaxFee | Private 🔐 |   | |
| └ | calculateLiquidityFee | Private 🔐 |   | |
| └ | calculateCharityFee | Private 🔐 |   | |
| └ | removeAllFee | Private 🔐 | 🛑  | |
| └ | restoreAllFee | Private 🔐 | 🛑  | |
| └ | isExcludedFromFee | Public ❗️ |   |NO❗️ |
| └ | _approve | Private 🔐 | 🛑  | |
| └ | _transfer | Private 🔐 | 🛑  | |
| └ | swapAndLiquify | Private 🔐 | 🛑  | lockTheSwap |
| └ | _swapForCharity | Private 🔐 | 🛑  | lockTheSwap |
| └ | swapTokensForEth | Private 🔐 | 🛑  | |
| └ | addLiquidity | Private 🔐 | 🛑  | |
| └ | _tokenTransfer | Private 🔐 | 🛑  | |
| └ | _transferStandard | Private 🔐 | 🛑  | |
| └ | _transferToExcluded | Private 🔐 | 🛑  | |
| └ | _transferFromExcluded | Private 🔐 | 🛑  | |
| └ | setDonating | External ❗️ | 🛑  | onlyOwner |
| └ | setDonatingForArray | External ❗️ | 🛑  | onlyOwner |
| └ | getDonating | Public ❗️ |   |NO❗️ |
```

Legend

```
|  Symbol  |  Meaning  |
|:--------:|-----------|
|    🛑    | Function can modify state |
|    💵    | Function is payable |
```

7. Tree description

```
(Attached as separate document for brevity)
```

MythX

Analysis e9e1f300-e215-4ff1-8f3e-090fd62e54bb

MythX

Started	Thu Jul 08 2021 20:12:29 GMT+0000 (Coordinated Universal Time)
Finished	Thu Jul 08 2021 20:14:07 GMT+0000 (Coordinated Universal Time)
Mode	Quick
Client Tool	Mythx-Vscode-Extension
Main Source File	/Contracts/Equitas.Sol

DETECTED VULNERABILITIES

HIGH	MEDIUM	LOW
0	20	2

ISSUES

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "name" is marked "public". However, it is never directly called by another function in the same contract or in any of its descendants. Consider to mark it as "external" instead.

Source file

/contracts/equitas.sol

Locations

```
return tokenFromReflection(_rOwned[account]);
}

function transfer(address recipient, uint256 amount)
public
override
returns (bool)
{
```

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "symbol" is marked "public". However, it is never directly called by another function in the same contract or in any of its descendants. Consider to mark it as "external" instead.

Source file

`/contracts/equitas.sol`

Locations

```
function transfer(address recipient, uint256 amount)
public
override
returns (bool)
{
_transfer(_msgSender(), recipient, amount);
return true;
return true;
}

```

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "decimals" is marked "public". However, it is never directly called by another function in the same contract or in any of its descendants. Consider to mark it as "external" instead.

Source file

`/contracts/equitas.sol`

Locations

```
{
_transfer(_msgSender(), recipient, amount);
return true;
}

function allowance(address owner, address spender)
public
view
override
```

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "totalSupply" is marked "public". However, it is never directly called by another function in the same contract or in any of its descendants. Consider to mark it as "external" instead.

Source file

/contracts/equitas.sol

Locations

```

function allowance(address owner, address spender)
public
view
override
returns (uint256)
{
return _allowances[owner][spender];
}

```

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "transfer" is marked "public". However, it is never directly called by another function in the same contract or in any of its descendants. Consider to mark it as "external" instead.

Source file

/contracts/equitas.sol

Locations

```
{
_approve(_msgSender(), spender, amount);
return true;
}

function transferFrom(
address sender,
address recipient,
uint256 amount
) public override returns (bool) {
_transfer(sender, recipient, amount);
_approve(
sender,
```

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "allowance" is marked "public". However, it is never directly called by another function in the same contract or in any of its descendants. Consider to mark it as "external" instead.

Source file

/contracts/equitas.sol

Locations

```
uint256 amount
) public override returns (bool) {
_transfer(sender, recipient, amount);

_approve(
sender,
_msgSender(),
_allowances[sender][_msgSender()].sub(
amount /*, "ERC20: transfer amount exceeds allowance"*/
)
);
```

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "approve" is marked "public". However, it is never directly called by another function in the same contract or in any of its descendants. Consider to mark it as "external" instead.

Source file

/contracts/equitas.sol

Locations

```
_msgSender(),
_allowances[sender][_msgSender()].sub(
amount /*, "ERC20: transfer amount exceeds allowance"*/

)
);
return true;
}

function increaseAllowance(address spender, uint256 addedValue)
public
virtual
returns (bool)
{
_approve(
_msgSender(),
spender,
```

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "transferFrom" is marked "public". However, it is never directly called by another function in the same contract or in any of its descendants. Consider to mark it as "external" instead.

Source file

/contracts/equitas.sol

Locations

```
returns (bool)
{
_approve(
_msgSender(),
spender,
_allowances[_msgSender()][spender].add(addedValue)
);
return true;
}

function decreaseAllowance(address spender, uint256 subtractedValue)
public
virtual
returns (bool)
{
_approve(
_msgSender(),
spender,
_allowances[_msgSender()][spender].sub(
subtractedValue /*, "ERC20: decreased allowance below zero"*/
)
);
```

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "increaseAllowance" is marked "public". However, it is never directly called by another function in the same contract or in any of its descendants. Consider to mark it as "external" instead.

Source file

/contracts/equitas.sol

Locations

```
spender,
_allowances[_msgSender()][spender].sub(
subtractedValue /*, "ERC20: decreased allowance below zero"*/
)
);
return true;
}

function isExcludedFromReward(address account) public view returns (bool) {
return _isExcluded[account];
}

function totalFees() public view returns (uint256) {
return _tFeeTotal;
}
```

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "decreaseAllowance" is marked "public". However, it is never directly called by another function in the same contract or in any of its descendants. Consider to mark it as "external" instead.

Source file

/contracts/equitas.sol

Locations

```

function totalFees() public view returns (uint256) {
return _tFeeTotal;
}

function deliver(uint256 tAmount) public {
address sender = _msgSender();
require(
!_isExcluded[sender],
"Excluded addresses cannot call this function"
);
uint256 rAmount = _getValues(tAmount).rAmount;

_rOwned[sender] = _rOwned[sender].sub(rAmount);
_rTotal = _rTotal.sub(rAmount);
_tFeeTotal = _tFeeTotal.add(tAmount);
}
```

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "isExcludedFromReward" is marked "public". However, it is never directly called by another function in the same contract or in any of its descendants. Consider to mark it as "external" instead.

Source file

/contracts/equitas.sol

Locations

```

_rOwned[sender] = _rOwned[sender].sub(rAmount);
_rTotal = _rTotal.sub(rAmount);
_tFeeTotal = _tFeeTotal.add(tAmount);
}

function reflectionFromToken(uint256 tAmount, bool deductTransferFee)
public
view
```

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "totalFees" is marked "public". However, it is never directly called by another function in the same contract or in any of its descendants. Consider to mark it as "external" instead.

Source file

/contracts/equitas.sol

Locations

```
}

function reflectionFromToken(uint256 tAmount, bool deductTransferFee)
public
view
returns (uint256)
{
require(tAmount <= _tTotal, "Amount must be less than supply");
if (!deductTransferFee) {
return _getValues(tAmount).rAmount;
```

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "deliver" is marked "public". However, it is never directly called by another function in the same contract or in any of its descendants. Consider to mark it as "external" instead.

Source file

/contracts/equitas.sol

Locations

```
returns (uint256)
{
require(tAmount <= _tTotal, "Amount must be less than supply");
if (!deductTransferFee) {
return _getValues(tAmount).rAmount;
} else {
return _getValues(tAmount).rTransferAmount;
}
}

function tokenFromReflection(uint256 rAmount)
public
view
returns (uint256)
{
require(
rAmount <= _rTotal,
"Amount must be less than total reflections"
);
uint256 currentRate = _getRate();
```

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "reflectionFromToken" is marked "public". However, it is never directly called by another function in the same contract or in any of its descendants. Consider to mark it as "external" instead.

Source file

/contracts/equitas.sol

Locations

```
require(
rAmount <= _rTotal,
"Amount must be less than total reflections"
);
uint256 currentRate = _getRate();
return rAmount.div(currentRate);
}

function excludeFromReward(address account) public onlyOwner() {
// require(account != 0x7a250d5630B4cF539739dF2C5dAcb4c659F2488D, 'We can not exclude Pancake router.');
require(!_isExcluded[account], "Account is already excluded");
if (_rOwned[account] > 0) {
_tOwned[account] = tokenFromReflection(_rOwned[account]);
```

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "excludeFromReward" is marked "public". However, it is never directly called by another function in the same contract or in any of its descendants. Consider to mark it as "external" instead.

Source file

/contracts/equitas.sol

Locations

```

function includeInReward(address account) external onlyOwner() {
require(_isExcluded[account], "Account is already excluded");
for (uint256 i = 0; i < _excluded.length; i++) {
if (_excluded[i] == account) {
_excluded[i] = _excluded[_excluded.length - 1];
_tOwned[account] = 0;
_isExcluded[account] = false;
_excluded.pop();
break;
}
}
}

function _transferBothExcluded(
address sender,
address recipient,

uint256 tAmount
) private {
```

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "excludeFromFee" is marked "public". However, it is never directly called by another function in the same contract or in any of its descendants. Consider to mark it as "external" instead.

Source file

/contracts/equitas.sol

Locations

```

function setNumTokensSellToAddToLiquidity(uint256 _numTokensSellToAddToLiquidity) external onlyOwner() {
numTokensSellToAddToLiquidity = _numTokensSellToAddToLiquidity;
}

function setNumTokensSellForCharity(uint256 _numTokensSellForCharity) external onlyOwner() {
numTokensSellForCharity = _numTokensSellForCharity;
}
```

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "includeInFee" is marked "public". However, it is never directly called by another function in the same contract or in any of its descendants. Consider to mark it as "external" instead.

Source file

/contracts/equitas.sol

Locations

```
}

function setNumTokensSellForCharity(uint256 _numTokensSellForCharity) external onlyOwner() {
numTokensSellForCharity = _numTokensSellForCharity;
}

function setCharityFeePercent(uint256 charityFee) external onlyOwner() {
```

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "setSwapAndLiquifyEnabled" is marked "public". However, it is never directly called by another function in the same contract or in any of its descendants. Consider to mark it as "external" instead.

Source file

/contracts/equitas.sol

Locations

```
private
view
returns (ValuesStruct memory)
{
(uint256 tTransferAmount, uint256 tFee, uint256 tLiquidity, uint256 tCharity) =
_getTValues(tAmount);

```

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "setSwapForCharityEnabled" is marked "public". However, it is never directly called by another function in the same contract or in any of its descendants. Consider to mark it as "external" instead.

Source file

/contracts/equitas.sol

Locations

```
returns (ValuesStruct memory)
{
(uint256 tTransferAmount, uint256 tFee, uint256 tLiquidity, uint256 tCharity) =
_getTValues(tAmount);

RValuesStruct memory rValuesStruct = RValuesStruct(tAmount, tFee, tLiquidity, tCharity, _getRate());
(uint256 rAmount, uint256 rTransferAmount, uint256 rFee, uint256 rCharity) =
_getRValues(rValuesStruct);
```

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "isExcludedFromFee" is marked "public". However, it is never directly called by another function in the same contract or in any of its descendants. Consider to mark it as "external" instead.

Source file

/contracts/equitas.sol

Locations

```
address from,
address to,
uint256 amount
) private {
require(from != address(0), "ERC20: transfer from the zero address");
require(to != address(0), "ERC20: transfer to the zero address");
require(amount > 0, "Transfer amount must be greater than zero");
if (
```

LOW

SWC-134

Call with hardcoded gas amount.

The highlighted function call forwards a fixed amount of gas. This is discouraged as the gas cost of EVM instructions may change in the future, which could break this contract's assumptions. If this was done to prevent reentrancy attacks, consider alternative methods such as the checks-effects-interactions pattern or reentrancy locks instead.

Source file

/contracts/equitas.sol

Locations

```
}

function recoverGas() external onlyOwner() {
(payable (owner())).transfer(address(this).balance);
}

```

LOW

SWC-134

Call with hardcoded gas amount.

The highlighted function call forwards a fixed amount of gas. This is discouraged as the gas cost of EVM instructions may change in the future, which could break this contract's assumptions. If this was done to prevent reentrancy attacks, consider alternative methods such as the checks-effects-interactions pattern or reentrancy locks instead.

Source file

`/contracts/equitas.sol`

Locations

```
function decimals() public view returns (uint8) {
return _decimals;
}

function totalSupply() public view override returns (uint256) {
return _tTotal;
}
```