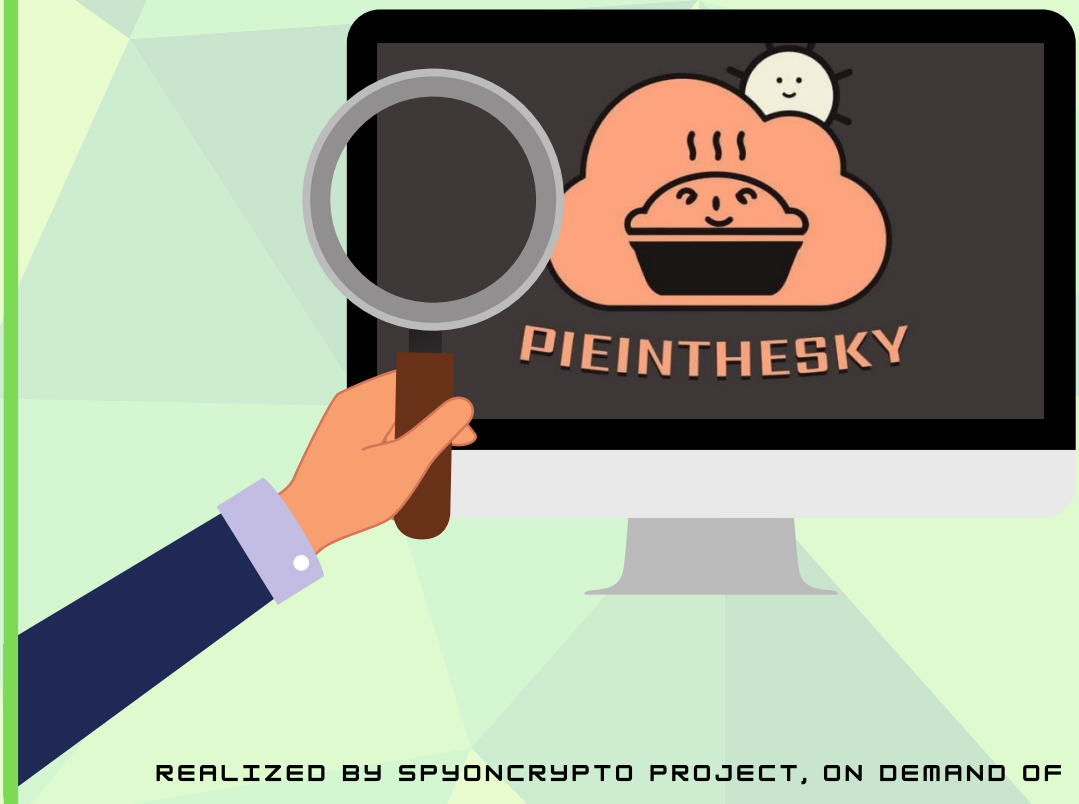
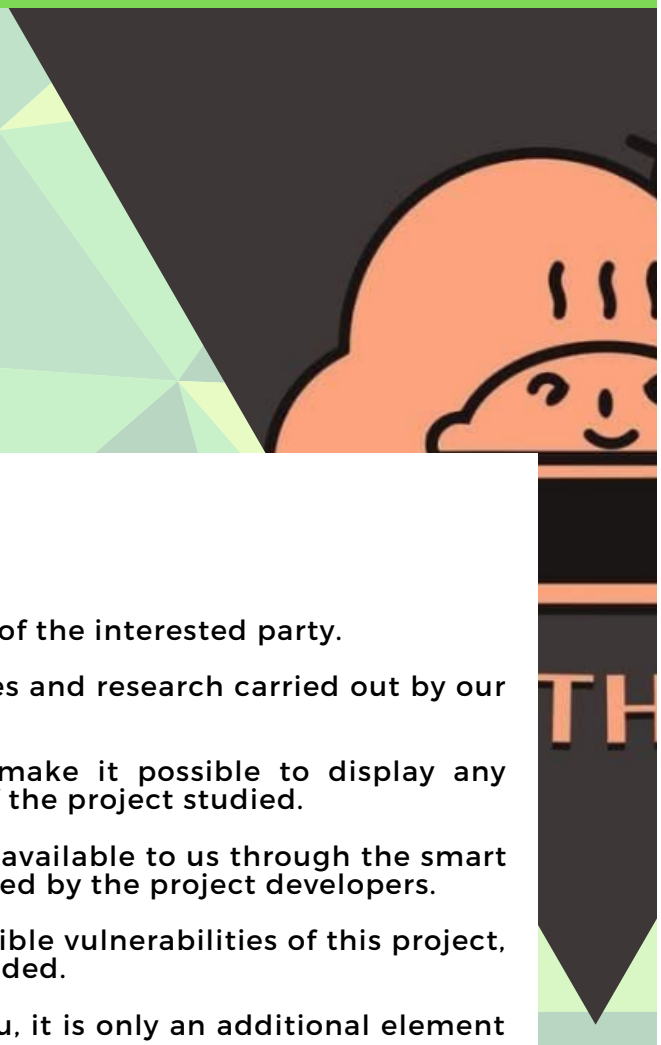
SPY ON CRYPTO

AUDIT PITS



REALIZED BY SPYONCRYPTO PROJECT, ON DEMAND OF PITS TEAM

DISCLAIMER



This file is an audit carried out at the request of the interested party.

This report is based on a multitude of analyses and research carried out by our team according to a predefined scheme.

The various steps set out in this file will make it possible to display any vulnerabilities relating to the cybersecurity of the project studied.

These searches are based on the information available to us through the smart contract, but also through information provided by the project developers.

In order to have a better overview of the possible vulnerabilities of this project, the complete reading of this file is recommended.

However, even if this report is available to you, it is only an additional element that can help you in your investigations.

Although a great deal of background work has been done in our investigations, we may have missed some elements, so further research on your part is necessary and advisable.

The conditions mentioned above in the disclaimer are not optional, so if you are not satisfied with them, we strongly urge you to stop reading and analyzing this file and to destroy any copies you have downloaded and/or printed.

These analyses and conclusions are not intended as investment advice. SpyonCrypto is not responsible for any loss of capital, which you are the only owner of.

This report is provided to you as, and without any conditions guaranteed.

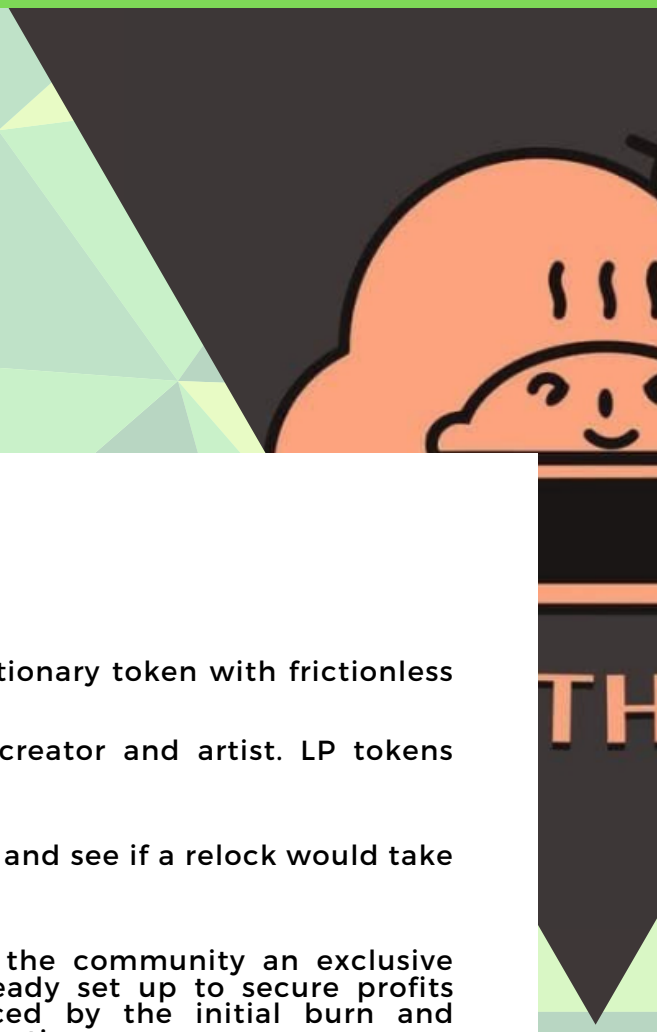
SpyonCrypto disclaims any and all liability to the law for any claim or demand by you or any other person for damages.

The analysis of the security is purely based on the smart contracts alone. No applications or operations were reviewed for security.

No product code has been reviewed.

SUMMARY

PRESENTATION



$PITS means PIE IN THE SKY. It is a deflationary token with frictionless TXN's. Based on NFT and Portofolio.

Physical NTFs being worked on by the creator and artist. LP tokens locked until October 22, 2021.

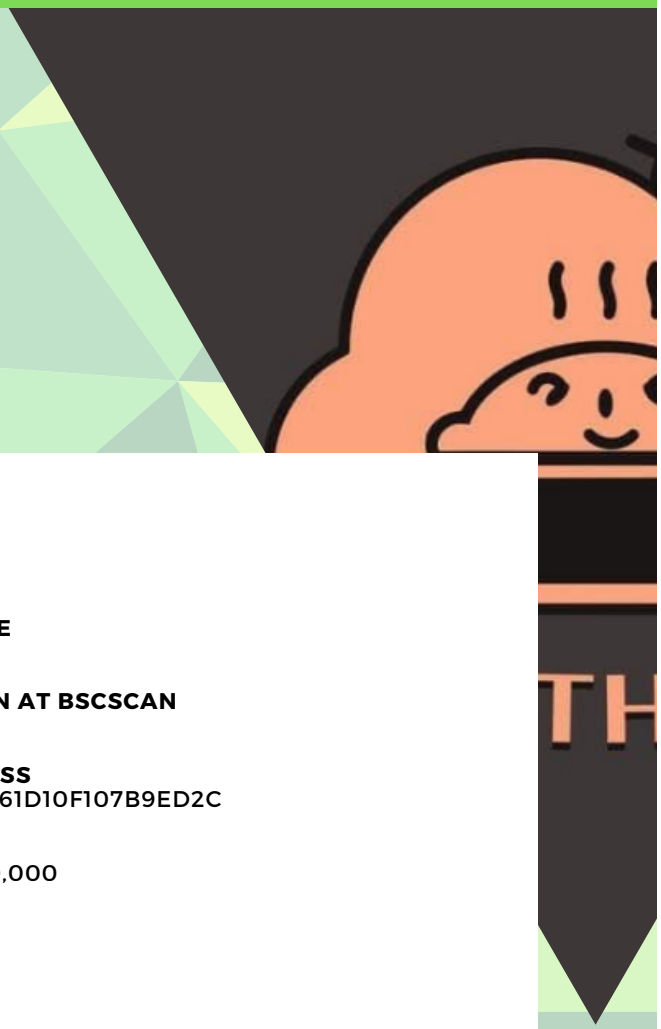
At this point we would look at the project and see if a relock would take place.

They burned half the supply to give at the community an exclusive chance to purchase tokens that are already set up to secure profits based on a deflationary aspect produced by the initial burn and continuous burn that happens every transaction.

Some important points :

- initial burn ; 23,750,000,000,000,000
- circulation supply ; 26,250,000,000,000,000
- deflationary ; 2,5% back to holders/2,5% sent straight to the oven.
- Holder structure shows burn address in 1st position, 1 external wallet with 7% of the supply and Pancakeswap in 3th position
- Ownership has not been renounced yet
- Fixed tx fee of 5%
- 53% of total supply have been burned

CONTRACT DETAILS



CONTRACT NAME
PIEINTHESKY

SUBMITTED FOR VERIFICATION AT BSCSCAN
2021-05-20

CONTRACT ADDRESS
0XCB0E64FF52B80F2C315C6C0A761D10F107B9ED2C

TOTAL SUPPLY
50,000,000,000,000,000

TOKEN TICKER
PITS

DECIMALS
9

TOKEN HOLDERS
506

TRANSACTIONS COUNT
2991

TOP 100 HOLDERS DOMINANCE
92,52%

CONTRACT DEPLOYER ADDRESS
0XE5759EDA2C122B917A3A6EE5B49A36F1DCD69144

CONTRACT'S CURRENT OXNER ADDRESS
0X0000000000000000000000000000000000000000

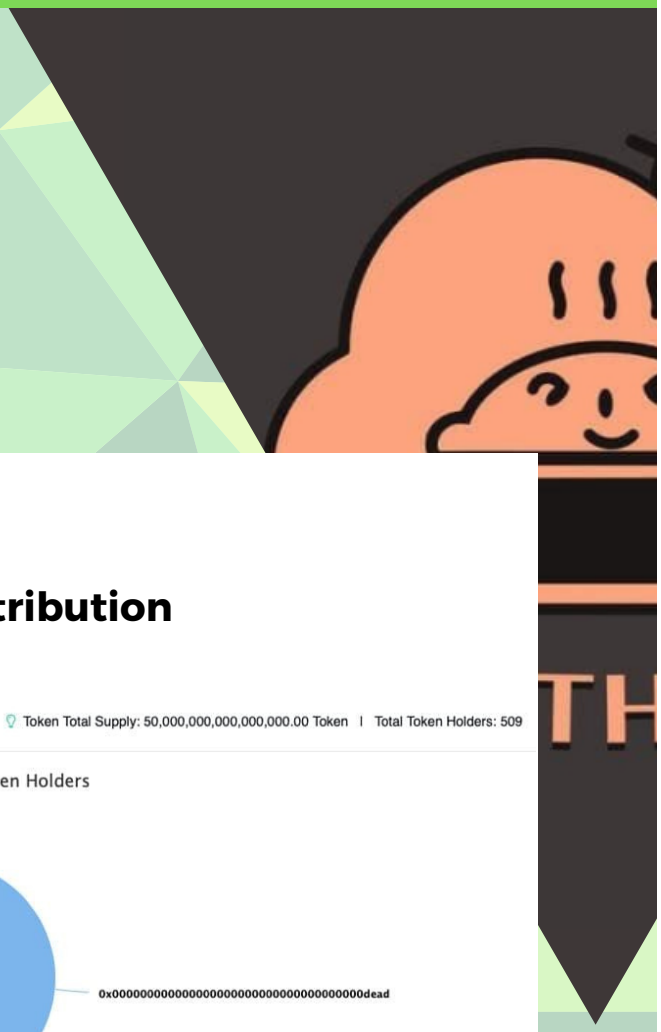
CURRENT LIQUIDITY FEE
5%

CURRENT TAX FEE
5%

TOTAL FEES
1041108396566017413415735O

DEPLOYED AT TRANSACTION
0XA913C052B8F6C0DEF7AE6D7C46E347892CEEDB69AF2DEBBC3DD13C9E
B290A37B

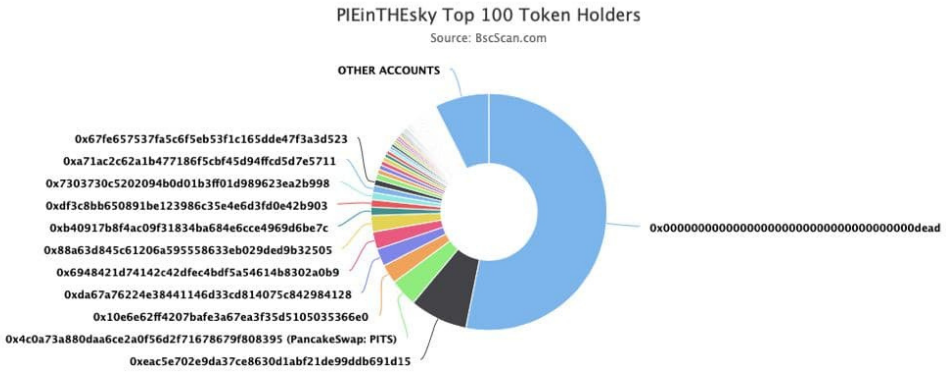
GRAPHIC ANALYSIS



"PITS" Token distribution

The top 100 holders collectively own 92.52% (46,262,372,850,144,100.00 Tokens) of PIEinTHEsky

Token Total Supply: 50,000,000,000,000,000.00 Token | Total Token Holders: 509

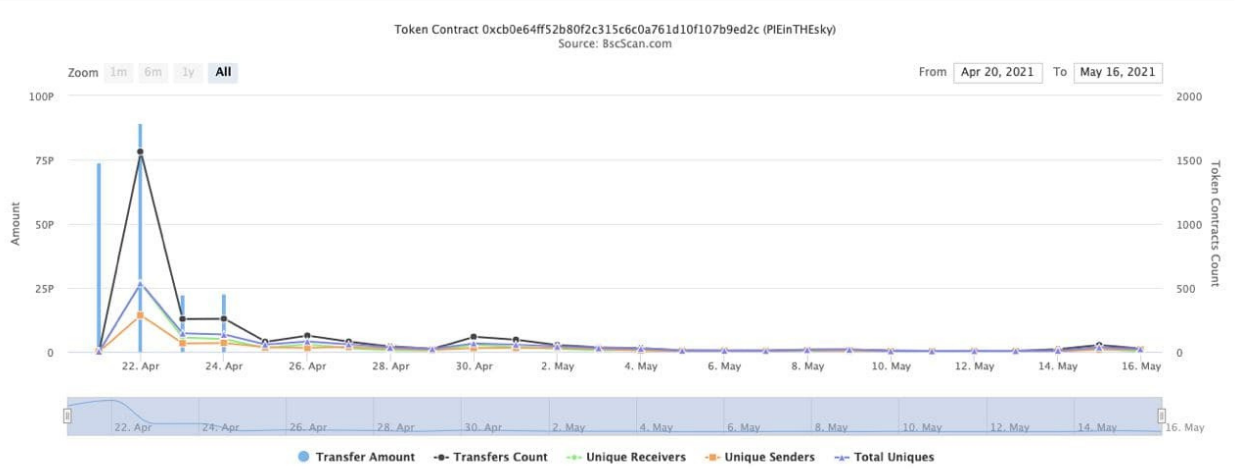


(A total of 46,262,372,850,144,100.00 tokens held by the top 100 accounts from the total supply of 50,000,000,000,000,000.00 token)

"PITS" contract interaction details

Time Series: Token Contract Overview

Wed 21, Apr 2021 - Sun 16, May 2021



DETECTED VULNERABILITIES

0

20

1

SECURITY ISSUES

MEDIUM SWC-000

Function could be marked as external.
The function definition of "owner" 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.

```
386       */
387      function owner() public view returns (address) {
388          return _owner;
389      }
```

```
405       */
406      function renounceOwnership() public virtual onlyOwner {
407          emit OwnershipTransferred(_owner, address(0));
408          _owner = address(0);
409      }
```

```
414      */
415     function transferOwnership(address newOwner) public virtual onlyOwner {
416         require(newOwner != address(0), "Ownable: new owner is the zero address");
417         emit OwnershipTransferred(_owner, newOwner);
418         _owner = newOwner;
419     }
420 }
```

```
448
449      function name() public view returns (string memory) {
450          return _name;
451      }
```

```
452
453      function symbol() public view returns (string memory) {
454          return _symbol;
455      }
```

```
456
457       function decimals() public view returns (uint8) {
458           return _decimals;
459       }
```

```
460
461       function totalSupply() public pure override returns (uint256) {
462           return _tTotal;
463       }
```

```
464
465       function balanceOf(address account) public view override returns (uint256) {
466           if (_isExcluded[account]) return _tOwned[account];
467           return tokenFromReflection(_rOwned[account]);
468       }
```

```
469
470       function transfer(address recipient, uint256 amount) public override returns (bool) {
471           _transfer(_msgSender(), recipient, amount);
472           return true;
473       }
```

```
474
475       function allowance(address owner, address spender) public view override returns (uint256) {
476           return _allowances[owner][spender];
477       }
```

```
478
479       function approve(address spender, uint256 amount) public override returns (bool) {
480           _approve(_msgSender(), spender, amount);
481           return true;
482       }
```

```
483
484       function transferFrom(address sender, address recipient, uint256 amount) public override returns (bool) {
485           _transfer(sender, recipient, amount);
486           _approve(sender, _msgSender(), _allowances[sender][_msgSender()].sub(amount, "ERC20: transfer amount exceeds allowance"));
487           return true;
488       }
```

```
489
490       function increaseAllowance(address spender, uint256 addedValue) public virtual returns (bool) {
491           _approve(_msgSender(), spender, _allowances[_msgSender()][spender].add(addedValue));
492           return true;
493       }
```

```
494
495     function decreaseAllowance(address spender, uint256 subtractedValue) public virtual returns (bool) {
496         _approve(_msgSender(), spender, _allowances[_msgSender()][spender].sub(subtractedValue, "ERC20: decreased allowance below zero"));
497         return true;
498     }
```

```
499
500       function isExcluded(address account) public view returns (bool) {
501           return _isExcluded[account];
502       }
```

```
503
504       function totalFees() public view returns (uint256) {
505           return _tFeeTotal;
506       }
```

```
507
508       function reflect(uint256 tAmount) public {
509           address sender = _msgSender();
510           require(!_isExcluded[sender], "Excluded addresses cannot call this function");
511           (uint256 rAmount,,,,) = _getValues(tAmount);
512           _rOwned[sender] = _rOwned[sender].sub(rAmount);
513           _rTotal = _rTotal.sub(rAmount);
514           _tFeeTotal = _tFeeTotal.add(tAmount);
515       }
```

```
516
517     function reflectionFromToken(uint256 tAmount, bool deductTransferFee) public view returns(uint256) {
518         require(tAmount <= _tTotal, "Amount must be less than supply");
519         if (!deductTransferFee) {
520             (uint256 rAmount,,,,) = _getValues(tAmount);
521             return rAmount;
522         } else {
523             (,uint256 rTransferAmount,,,) = _getValues(tAmount);
524             return rTransferAmount;
525         }
526     }
```

MEDIUM SWC-128

Loop over unbounded data structure.
Gas consumption in function "includeAccount" in contract "PIEinTHEsky" depends on the size of data structures or values that may grow unboundedly. If the data structure grows too large, the gas required to execute the code will exceed the block gas limit, effectively causing a denial-of-service condition. Consider that an attacker might attempt to cause this condition on purpose.

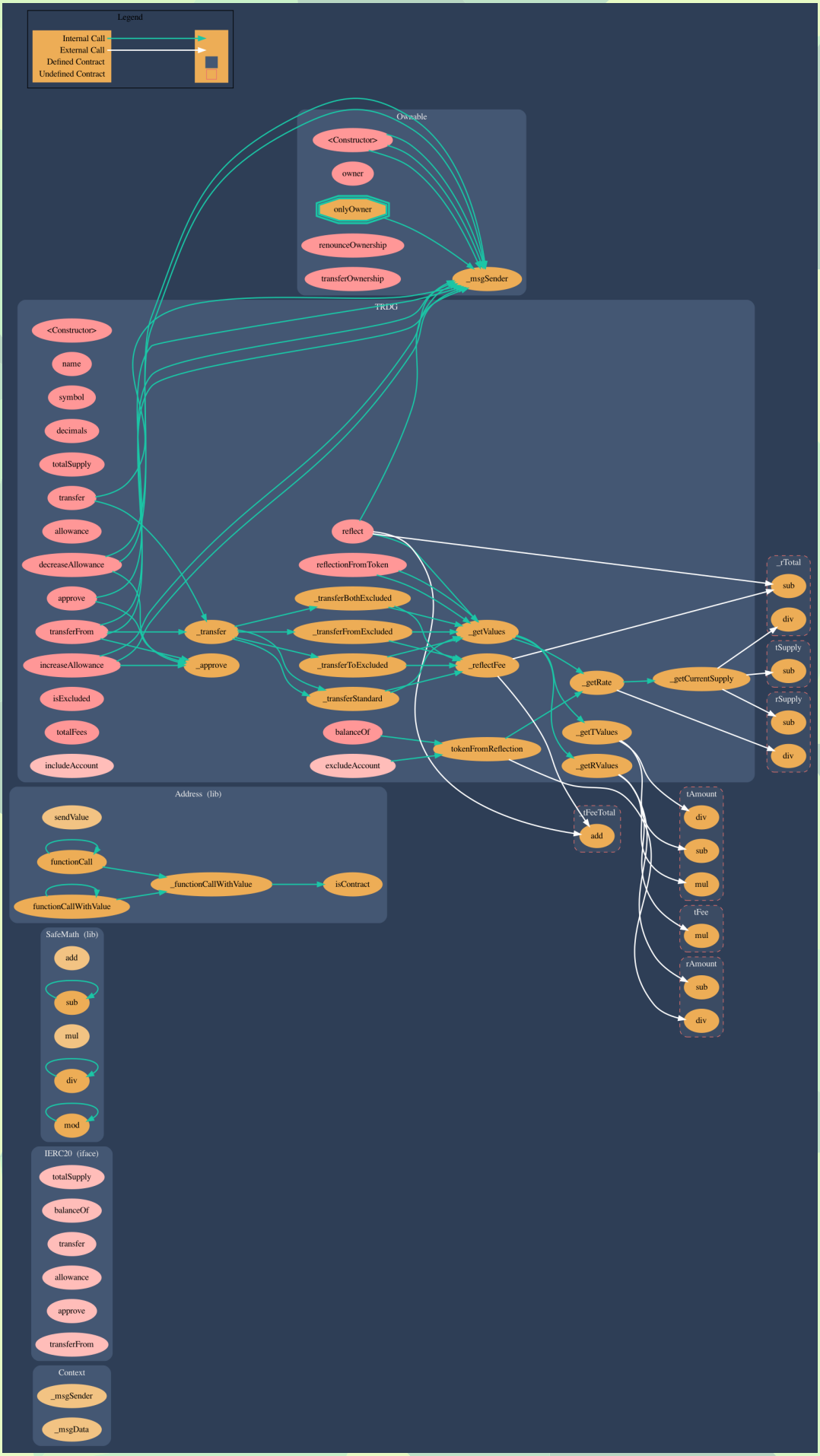
```
542
543     function includeAccount(address account) external onlyOwner() {
544         require(_isExcluded[account], "Account is already excluded");
545         for (uint256 i = 0; i < _excluded.length; i++) {
546             if (_excluded[i] == account) {
547                 _excluded[i] = _excluded[_excluded.length - 1];
548                 _tOwned[account] = 0;
549                 _isExcluded[account] = false;
550                 _excluded.pop();
551                 break;
552             }
553         }
554     }
```

```
646
647     function _getCurrentSupply() private view returns(uint256, uint256) {
648         uint256 rSupply = _rTotal;
649         uint256 tSupply = _tTotal;
650         for (uint256 i = 0; i < _excluded.length; i++) {
651             if (_rOwned[_excluded[i]] > rSupply || _tOwned[_excluded[i]] > tSupply) return (_rTotal, _tTotal);
652             rSupply = rSupply.sub(_rOwned[_excluded[i]]);
653             tSupply = tSupply.sub(_tOwned[_excluded[i]]);
654         }
655         if (rSupply < _rTotal.div(_tTotal)) return (_rTotal, _tTotal);
656         return (rSupply, tSupply);
657     }
658 }
```

LOW SCWC-103

A floating pragma is set.
The current pragma Solidity directive is ""^0.7.6"".
It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

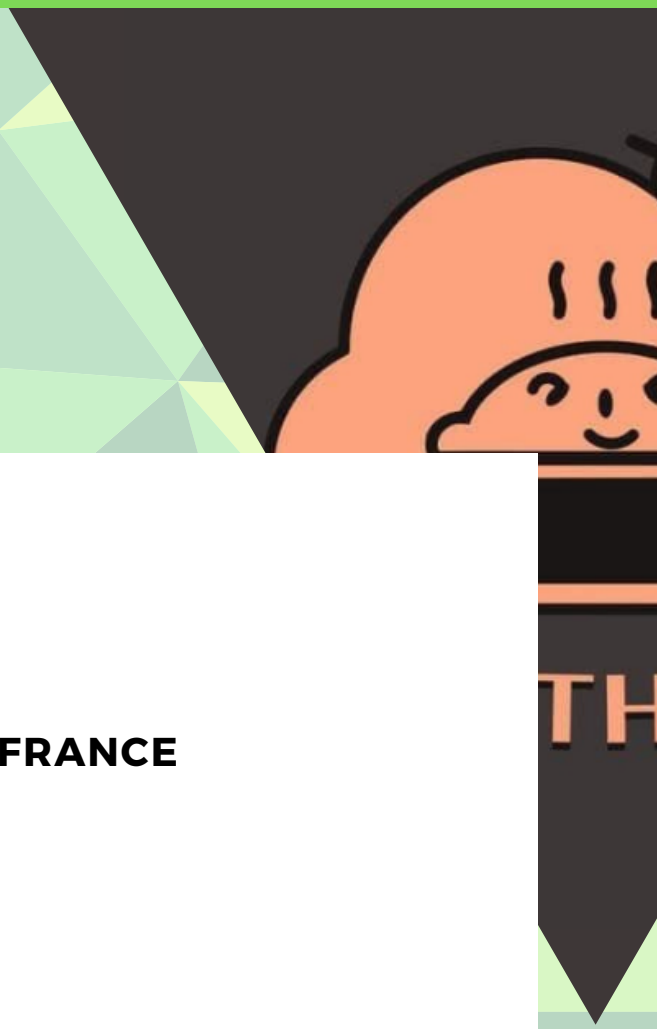
```
 5
 6  // https://t.me/PIEinTHEskyCHAT
 7
 8  pragma solidity ^0.7.6;
 9
10  abstract contract Context {
11      function _msgSender() internal view virtual returns (address payable) {
12          return msg.sender;
13      }
```


Legend
Internal Call
External Call
Defined Contract
Undefined Contract
Ownable
<Constructor>
owner
onlyOwner
renounceOwnership
transferOwnership
_msgSender
TRDG
<Constructor>
name
symbol
decimals
totalSupply
transfer
allowance
decreaseAllowance
approve
transferFrom
increaseAllowance
isExcluded
totalFees
includeAccount
_transfer
_approve
reflect
reflectionFromToken
_transferBothExcluded
_transferFromExcluded
_transferToExcluded
_transferStandard
balanceOf
excludeAccount
_getValues
_reflectFee
tokenFromReflection
_getRate
_getCurrentSupply
_getTValues
_getRValues
_rTotal
sub
div
tSupply
sub
rSupply
sub
div
tFeeTotal
add
tAmount
div
sub
mul
tFee
mul
tAmount
sub
div
Address (lib)
sendValue
functionCall
functionCallWithValue
_functionCallWithValue
isContract
SafeMath (lib)
add
sub
mul
div
mod
IERC20 (iface)
totalSupply
balanceOf
transfer
allowance
approve
transferFrom
Context
_msgSender
_msgData

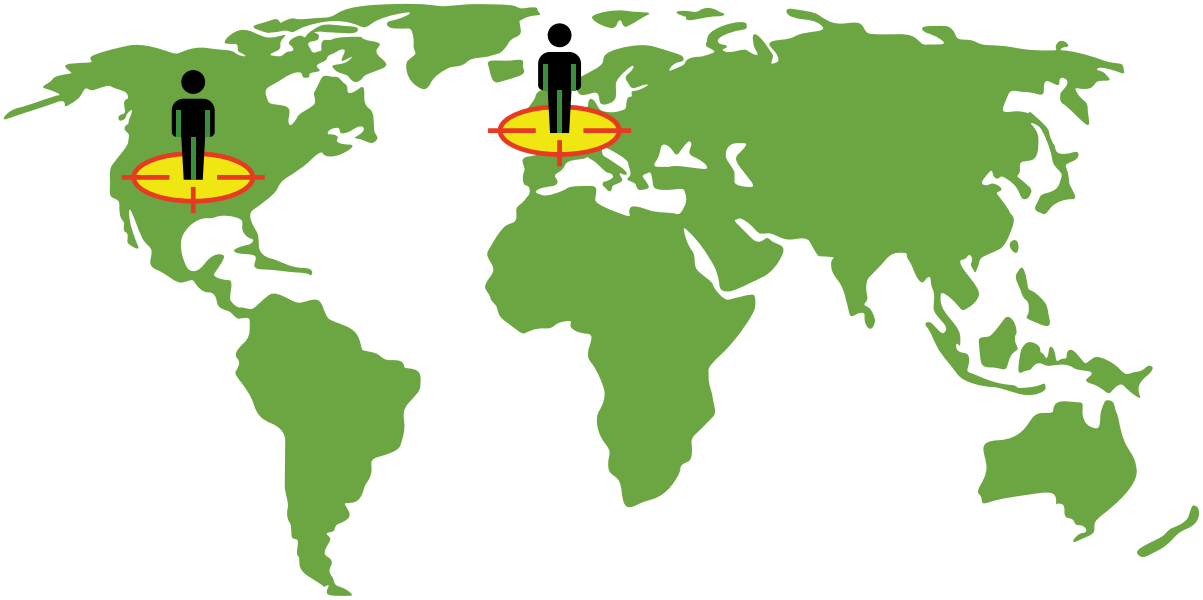
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	◆	NO !
Name	Public		NO !
Symbol	Public		NO !
Decimals	Public		NO !
TotalSupply	Public		NO !
BalanceOf	Public		NO !
Transfer	Public	◆	NO !
Allowance	Public		NO !
Approve	Public	◆	NO !
TransferFrom	Public	◆	NO !
IncreaseAllowance	Public	◆	NO !
DecreaseAllowance	Public	◆	NO !
IsExcluded	Public		NO !
TotalFees	Public		NO !
Reflect	Public	◆	NO !
ReflectionFromToken	Public		NO !
TokenFromReflection	Public		NO !
ExcludeAmount	External	◆	Only Owner
IncludeAmount	External	◆	Only Owner
_Approve	Private	◆	
_Transfer	Private	◆	
_TransferStandard	Private	◆	
_TransferToExcluded	Private	◆	
_TransferFromExcluded	Private	◆	
_TransferBothExcluded	Private	◆	
_ReflectFee	Private	◆	
_GetValues	Private		
_GetTValues	Private		
_GetRValues	Private		
_GetRate	Private		
_GetCurrentSupply	Private		

◆ = Function can modify state

LOCATION TEAM



TEAM USA AND FRANCE



SOCIAL MEDIA

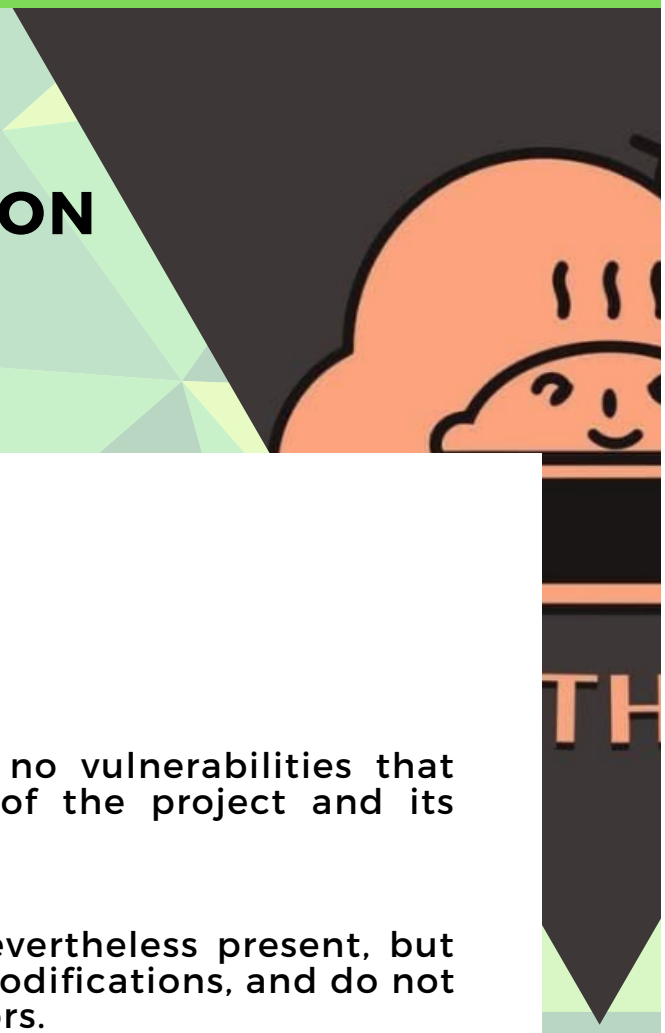
https://twitter.com/pieintheskybsc?s=21

https://t.me/PIEinTHEskyCHAT

https://youtube.com/channel/UCx7jttJvDLBHw6Qb9hrRzfA

https://www.pieinthesky.online/

NOTE AND CONCLUSION



The $PITS's smart contract has no vulnerabilities that would jeopardise the interests of the project and its investors.

Some "micro weaknesses" are nevertheless present, but they do not indicate a need for modifications, and do not change the security of the investors.

More marketing and partnership are advised in order to develop the project to reach a larger community.

$PITS has potential, now it has to be exploited.

TEAM DÉCISION

TG : SPYONCRYPTO
TG CHAT: SPYONCRYPTO CHAT
TWITTER: @spyoncrypto
MAIL: SOCIAL@SPYONCRYPTO.COM
SITE WEB: WWW.SPYONCRYPTO.COM