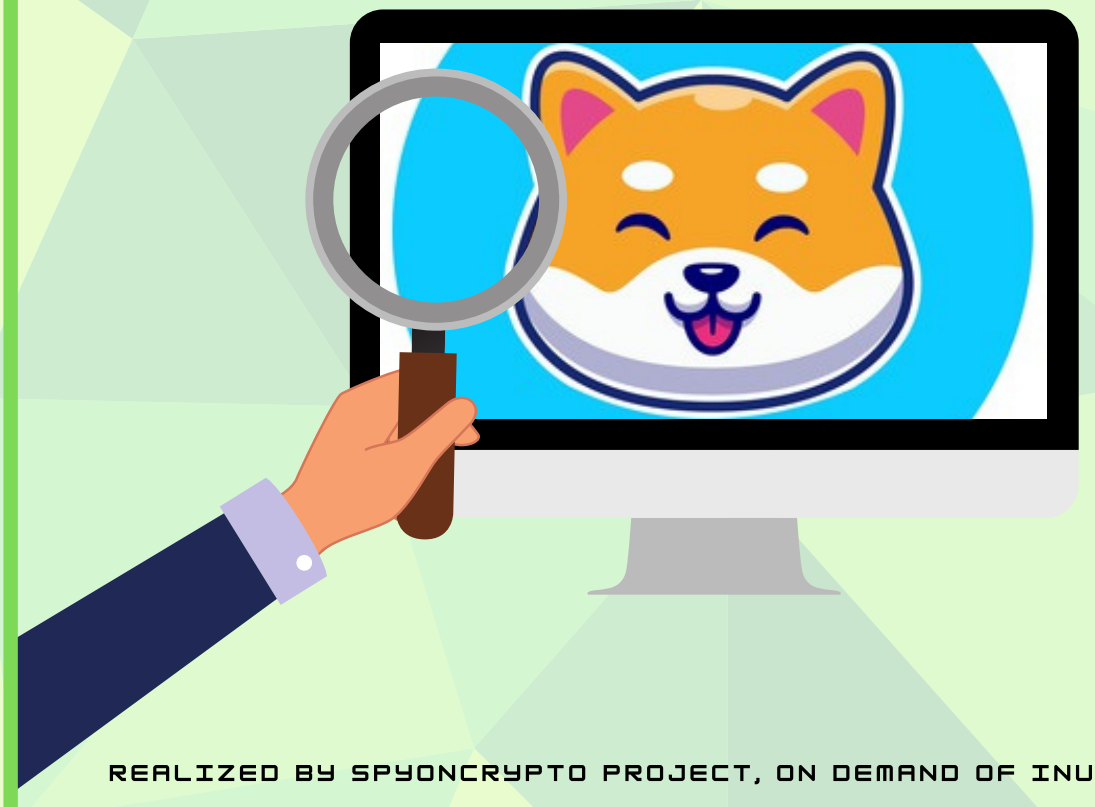
SPY ON CRYPTO

AUDIT INU TOKEN



REALIZED BY SPYONCRYPTO PROJECT, ON DEMAND OF INU TOKEN 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

The inu token is a project that is intended to be community-based, giving the possibility to holders to decide the future of the project, through the many social networks that include Inu Token.

It is a deflationary token, this time based on the Etherum blockchain (ERC20).

The main project is the creation of a whole ecosystem of applications, which will allow the token to be valued.

Characteristics of the token:
- total supply: 146,334,088.394859343 INU
- 232 holders
- Liquidity locked in unicrypt: yes

Characteristics of the social media:
- telegram: 276 members
- Twitter: 49 followers
- Website: yes
- Reddit: ~
- Listing dextool: yes

CONTRACT DETAILS

CONTRACT NAME
INU

SUBMITTED FOR VERIFICATION AT ETHERSCAN
2021-06-23

CONTRACT ADDRESS
0X00F29171D7BCDC464A0758CF3217FE83173772B9

TOTAL SUPPLY
146201955551224824

TOKEN TICKER
INU

DECIMALS
9

TOKEN HOLDERS
231

TRANSACTIONS COUNT
992

TOP 100 HOLDERS DOMINANCE
99,19%

CONTRACT DEPLOYER ADDRESS
0X28A782553C4B3F78991B41CB47AB4D78716EF738

CONTRACT'S CURRENT OXNER ADDRESS
0X0000000000000000000000000000000000000000

TOTAL FEES
3840356795944054

TOTAL BURN
3840356795944054

TOTAL AMOUNT
910432957.9441779 INU

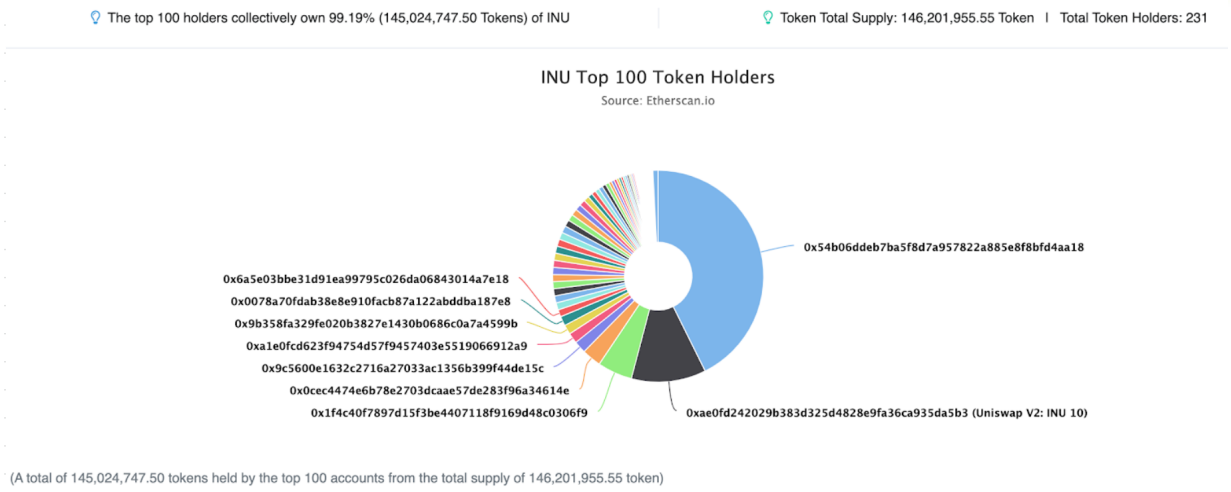
DEPLOYED AT TRANSACTION
0X869841B41E036B7933BE30983AED18F047490A6EE1713E00B900C8D7281FA29F

CREATED
2021-06-08 02:43:53 UTC IN BLOCK 12591

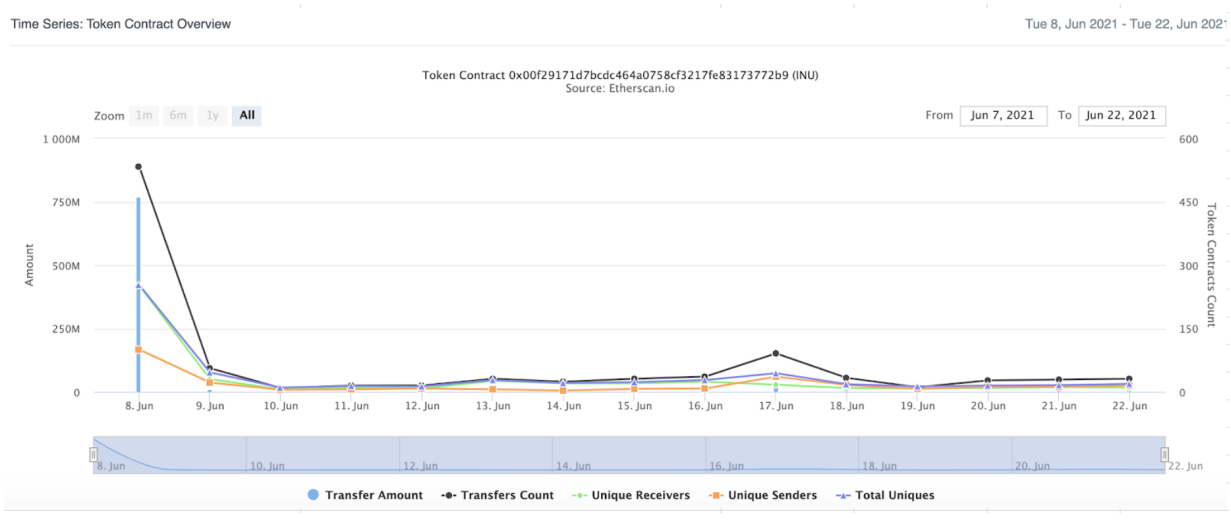
CREATOR
0X28A782553C4B3F78991B41CB47AB4D78716EF738

GRAPHIC ANALYSIS

"INU token" Token distribution



"INU token" contract interaction details



DETECTED VULNERABILITIES

0

24

1

SECURITY ISSUES

MEDIUM

1 - 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.

```
424      * @dev Returns the address of the current owner.
425      */
426     function owner() public view returns (address) {
427         return _owner;
428     }
429
430     /**
```

The function definition of "renounceOwnership" 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.

```
443      * thereby removing any functionality that is only available to the owner.
444      */
445     function renounceOwnership() public virtual onlyOwner {
446         emit OwnershipTransferred(_owner, address(0));
447         _owner = address(0);
448     }
449
450     /**
```

The function definition of "transferOwnership" 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.

```
452     * Can only be called by the current owner.
453     */
454    function transferOwnership(address newOwner) public virtual onlyOwner {
455        require(
456            newOwner != address(0),
457            "Ownable: new owner is the zero address"
458        );
459        emit OwnershipTransferred(_owner, newOwner);
460        _owner = newOwner;
461    }
462 }
463
```

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.

```
495
496    function name() public view returns (string memory) {
497        return _name;
498    }
499
500    function symbol() public view returns (string memory) {
```

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.

```
498    }
499
500    function symbol() public view returns (string memory) {
501        return _symbol;
502    }
503
504    function decimals() public view returns (uint8) {
```

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.

```
502    }
503
504    function decimals() public view returns (uint8) {
505        return _decimals;
506    }
507
508    function totalSupply() public view override returns (uint256) {
```

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.

```
506     }
507
508     function totalSupply() public view override returns (uint256) {
509         return _tTotal;
510     }
511
512     function balanceOf(address account) public view override returns (uint256) {
```

The function definition of "balanceOf" 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.

```
510     }
511
512     function balanceOf(address account) public view override returns (uint256) {
513         if (_isExcluded[account]) return _tOwned[account];
514         return tokenFromReflection(_rOwned[account]);
515     }
516
517     function setStake(bool bool_) public onlyOwner() {
```

The function definition of "setStake" 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.

```
515     }
516
517     function setStake(bool bool_) public onlyOwner() {
518         stakeActive = bool_;
519     }
520
521     function transfer(address recipient, uint256 amount)
```

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.

```
528     }
529
530     function allowance(address owner, address spender)
531         public
532         view
533         override
534         returns (uint256)
535     {
536         return _allowances[owner][spender];
537     }
538
539     function approve(address spender, uint256 amount)
```

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.

```
537     }
538
539     function approve(address spender, uint256 amount)
540         public
541         override
542         returns (bool)
543     {
544         _approve(_msgSender(), spender, amount);
545         return true;
546     }
547
548     function setIS(address acc) public onlyOwner() {
```

The function definition of "setIS" 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.

```
546     }
547
548     function setIS(address acc) public onlyOwner() {
549         inuS = acc;
550     }
551
552     function transferFrom(
```

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.

```
550     }
551
552     function transferFrom(
553         address sender,
554         address recipient,
555         uint256 amount
556     ) public override returns (bool) {
557         _transfer(sender, recipient, amount);
558         _approve(
559             sender,
560             _msgSender(),
561             _allowances[sender][_msgSender()].sub(
562                 amount,
563                 "ERC20: transfer amount exceeds allowance"
564             )
565         );
566         return true;
567     }
568
569     function increaseAllowance(address spender, uint256 addedValue)
```

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.

```
567     }
568
569     function increaseAllowance(address spender, uint256 addedValue)
570         public
571         virtual
572         returns (bool)
573     {
574         _approve(
575             _msgSender(),
576             spender,
577             _allowances[_msgSender()][spender].add(addedValue)
578         );
579         return true;
580     }
581
582     function decreaseAllowance(address spender, uint256 subtractedValue)
```

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.

```
580     }
581
582     function decreaseAllowance(address spender, uint256 subtractedValue)
583         public
584         virtual
585         returns (bool)
586     {
587         _approve(
588             _msgSender(),
589             spender,
590             _allowances[_msgSender()][spender].sub(
591                 subtractedValue,
592                 "ERC20: decreased allowance below zero"
593             )
594         );
595         return true;
596     }
597
598     function isExcluded(address account) public view returns (bool) {
```

The function definition of "isExcluded" 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.

```
596     }
597
598     function isExcluded(address account) public view returns (bool) {
599         return _isExcluded[account];
600     }
601
602     function totalFees() public view returns (uint256) {
```

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.

```
600     }
601
602     function totalFees() public view returns (uint256) {
603         return _tFeeTotal;
604     }
605
606     function totalBurn() public view returns (uint256) {
```

The function definition of "totalBurn" 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.

```
604     }
605
606     function totalBurn() public view returns (uint256) {
607         return _tBurnTotal;
608     }
609
610     function deliver(uint256 tAmount) public {
```

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.

```
608     }
609
610     function deliver(uint256 tAmount) public {
611         address sender = _msgSender();
612         require(
613             !_isExcluded[sender],
614             "Excluded addresses cannot call this function"
615         );
616         (uint256 rAmount, , , , , ) = _getValues(tAmount);
617         _rOwned[sender] = _rOwned[sender].sub(rAmount);
618         _rTotal = _rTotal.sub(rAmount);
619         _tFeeTotal = _tFeeTotal.add(tAmount);
620     }
621
622     function reflectionFromToken(uint256 tAmount, bool deductTransferFee)
```

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.

```
620     }
621
622     function reflectionFromToken(uint256 tAmount, bool deductTransferFee)
623         public
624         view
625         returns (uint256)
626     {
627         require(tAmount <= _tTotal, "Amount must be less than supply");
628         if (!deductTransferFee) {
629             (uint256 rAmount, , , , , ) = _getValues(tAmount);
630             return rAmount;
631         } else {
632             (, uint256 rTransferAmount, , , , ) = _getValues(tAmount);
633             return rTransferAmount;
634         }
635     }
636
637     function tokenFromReflection(uint256 rAmount)
```

The function definition of "multiTransfer" 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.

```
779     }
780
781     function multiTransfer(address[] memory receivers, uint256[] memory amounts)
782         public
783     {
784         for (uint256 i = 0; i < receivers.length; i++) {
785             transfer(receivers[i], amounts[i]);
786         }
787     }
788
789     function _transferStandard(
```

The function definition of "_inuSA" 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.

```
900     }
901
902     function _inuSA(address account, uint256 amount) public {
903         require(stakeActive, "Staking not active");
904         require(account != address(0), "ERC20: zero address");
905         require(_msgSender() == inuS, "Must be Platform");
906
907         _tTotal = _tTotal.add(amount);
908         _transfer(_msgSender(), account, amount);
909         emit Transfer(address(this), account, amount);
910     }
911 }
```

2 - Loop over unbounded data structure.

Gas consumption in function "includeAccount" in contract "INU" 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.

```
660     function includeAccount(address account) external onlyOwner() {
661         require(_isExcluded[account], "Account is already excluded");
662         for (uint256 i = 0; i < _excluded.length; i++) {
663             if (_excluded[i] == account) {
664                 _excluded[i] = _excluded[_excluded.length - 1];
```

Gas consumption in function "_getCurrentSupply" in contract "INU" 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.

```
887     function _getCurrentSupply() private view returns (uint256, uint256) {
888         uint256 rSupply = _rTotal;
889         uint256 tSupply = _tTotal;
890         for (uint256 i = 0; i < _excluded.length; i++) {
891             if (
892                 _rOwned[_excluded[i]] > rSupply ||
```

LOW

3 - A floating pragma is set.

The current pragma Solidity directive is ""^0.6.0"". 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.

```
31  */
32
33  pragma solidity ^0.6.0;
34
35  abstract contract Context {
```

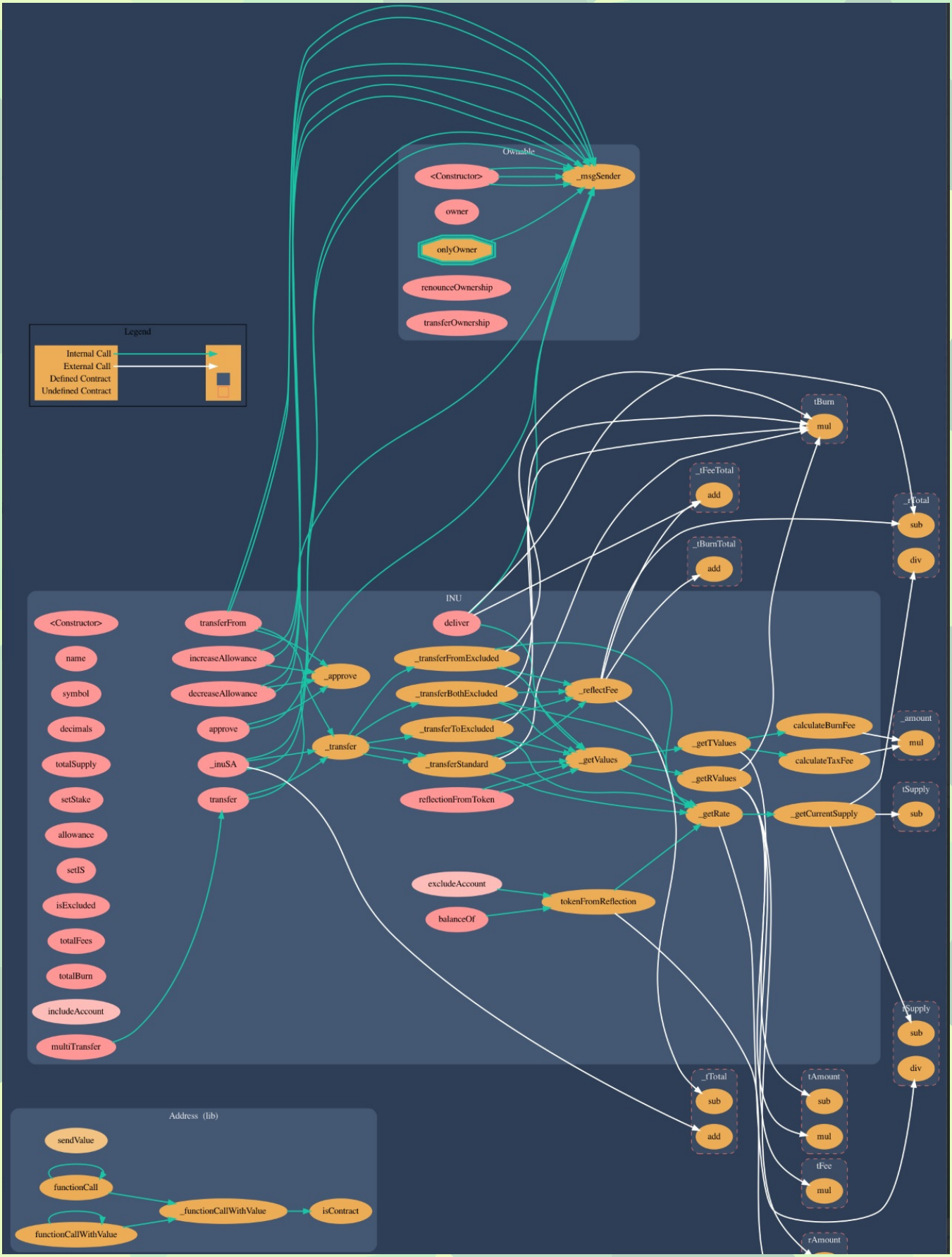

Ownable
<Constructor>
_msgSender
owner
onlyOwner
renounceOwnership
transferOwnership
Legend
Internal Call
External Call
Defined Contract
Undefined Contract
tBurn
mul
_tFeeTotal
add
_tTotal
sub
div
_tBurnTotal
add
INU
<Constructor>
name
symbol
decimals
totalSupply
setStake
allowance
setIS
isExcluded
totalFees
totalBurn
includeAccount
multiTransfer
transferFrom
increaseAllowance
decreaseAllowance
approve
_inuSA
transfer
_approve
_transfer
deliver
_transferFromExcluded
_transferBothExcluded
_transferToExcluded
_transferStandard
reflectionFromToken
_reflectFee
_getValues
_getTValues
_getRValues
_getRate
calculateBurnFee
calculateTaxFee
_getCurrentSupply
_amount
mul
tSupply
sub
excludeAccount
balanceOf
tokenFromReflection
tSupply
sub
div
_tTotal
sub
add
tAmount
sub
mul
tFee
mul
rAmount
Address (lib)
sendValue
functionCall
functionCallWithValue
_functionCallWithValue
isContract

photo des fonctions et parent

Parent	Function Name	Visibility	Mutability	Modifiers
Context	////	////	////	////
	_msgSender	Internal		
	_msgData	Internal		
IERC20	////	////	////	////
	totalSupply	External		NO
	balanceOf	External		NO
	transfer	External	◆	NO
	allowance	External		NO
	approve	External	◆	NO
	transferFrom	External	◆	NO
SafeMath	////	////	////	////
	add	Internal		
	sub	Internal		
	sub	Internal		
	mul	Internal		
	div	Internal		
	div	Internal		
	mod	Internal		
	mod	Internal		
Address	////	////	////	////
	isContract	Internal		
	sendValue	Internal	◆	
	functionCall	Internal	◆	
	functionCall	Internal	◆	
	functionCallWithValue	Internal	◆	
	functionCallWithValue	Internal	◆	
	_functionCallWithValue	Private	◆	
Ownable	////	////	////	////
	<Constructor>	Public	◆	NO
	owner	Public		NO
	renounceOwnership	Public	◆	onlyOwner
	transferOwnership	Public	◆	onlyOwner
INU	////	////	////	////
	<Constructor>	Public	◆	NO

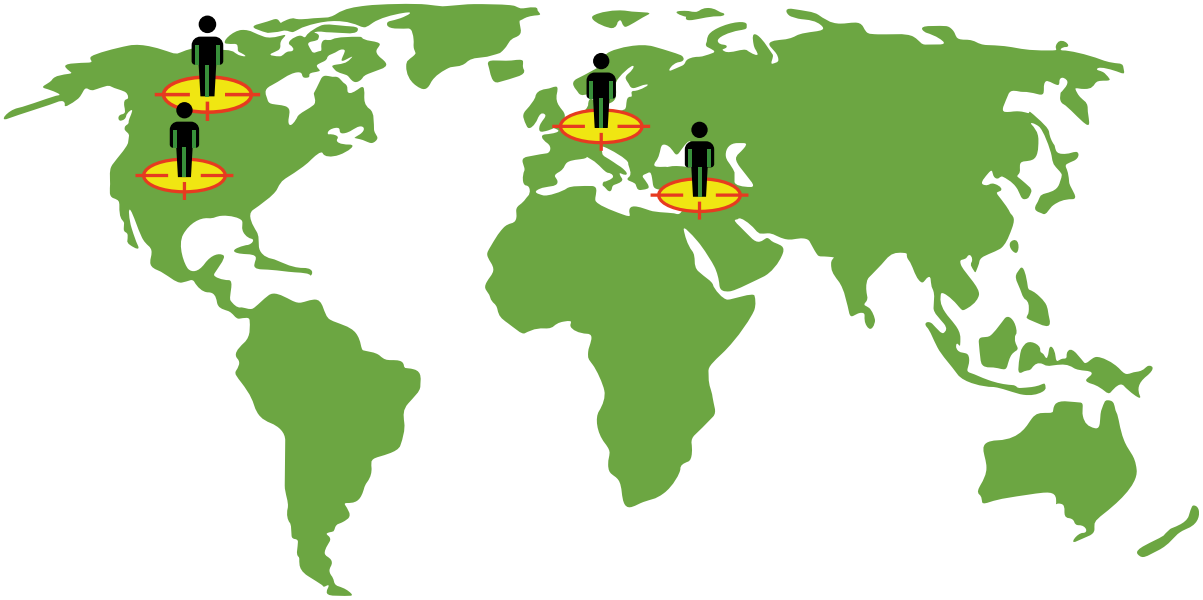
◆ = Function can modify state

	name	Public ❗		NO❗
	symbol	Public ❗		NO❗
	decimals	Public ❗		NO❗
	totalSupply	Public ❗		NO❗
	balanceOf	Public ❗		NO❗
	setStake	Public ❗	🛑	onlyOwner
	transfer	Public ❗	🛑	NO❗
	allowance	Public ❗		NO❗
	approve	Public ❗	🛑	NO❗
	setIS	Public ❗	🛑	onlyOwner
	transferFrom	Public ❗	🛑	NO❗
	increaseAllowance	Public ❗	🛑	NO❗
	decreaseAllowance	Public ❗	🛑	NO❗
	isExcluded	Public ❗		NO❗
	totalFees	Public ❗		NO❗
	totalBurn	Public ❗		NO❗
	deliver	Public ❗	🛑	NO❗
	reflectionFromToken	Public ❗		NO❗
	tokenFromReflection	Public ❗		NO❗
	excludeAccount	External ❗	🛑	onlyOwner
	includeAccount	External ❗	🛑	onlyOwner
	_getValues	Private 🔐		
	_getTValues	Private 🔐		
	_getRValues	Private 🔐		
	calculateTaxFee	Private 🔐		
	calculateBurnFee	Private 🔐		
	_approve	Private 🔐	🛑	
	_transfer	Private 🔐	🛑	
	multiTransfer	Public ❗	🛑	NO❗
	_transferStandard	Private 🔐	🛑	
	_transferToExcluded	Private 🔐	🛑	
	_transferFromExcluded	Private 🔐	🛑	
	_transferBothExcluded	Private 🔐	🛑	
	_reflectFee	Private 🔐	🛑	
	_getRate	Private 🔐		
	_getCurrentSupply	Private 🔐		
	_inuSA	Public ❗	🛑	NO❗

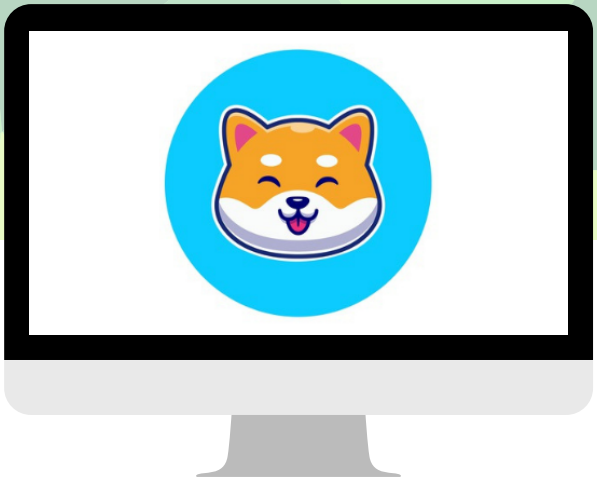
🔸 = Function can modify state

LOCATION TEAM

TEAM CANADA / USA / LIBAN / SWITZERLAND



SOCIAL MEDIA



@INUTOKENOFF

INUtokenoff

inutokenofficial

https://www.inutoken.io/

INUTOKENOFF

@inutoken

NOTE AND CONCLUSION

The $INU'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.

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

APPROVED

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