

<https://oracle-group.dev>

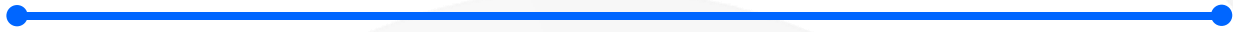


## AUDIT REPORT

### TRONX SOLIDITY CODE

HASH

URL



## Introduction

Checks The Contract Code For Security Vulnerabilities And Bad Practices

Transaction Origin: 'tx.origin' used

Pos: 257: Check-effects-interaction: Potential violation of Checks-Effects-Interaction pattern in withdraw(): Could potentially lead to re-entrancy vulnerability.



---

Inline Assembly: Inline assembly used

Block Timestamp: Can be influenced by miners

Low Level Calls: Should only be used by experienced devs

Block Hash: Can be influenced by miners

Selfdestruct: Contracts using destructed contract can be broken

Gas Costs: Too high gas requirement of functions

This On Local Calls: Invocation of local functions via 'this'

Delete Dynamic Array: Use require/assert to ensure complete deletion

For Loop Over Dynamic Array: Iterations depend on dynamic array's size

Ether Transfer In Loop: Transferring Ether in a for/while/do-while loop

TRC20: 'decimals' should be 'uint8'

Constant/View/Pure Functions: Potentially constant/view/pure functions

Similar Variable Names: Variable names are too similar

withdraw(): Variables have very similar names "pool\_bonuses" and "pool\_bonus"

userInfo(address): Variables have very similar names "pool\_bonuses" and "pool\_bonus"

userInfoTotals(address): Variables have very similar names "total\_deposited" and "total\_deposits"

No Return: Function with 'returns' not returning

Guard Conditions: Ensure appropriate use of require/assert

---

Use "assert(x)" if you never ever want x to be false, not in an y circumstance

(apart from a bug in your code). Use "require(x)" if x can be false, due to e.g. invalid input or a failing external component.

Pos: 138:61;143:28;

Result Not Used: The result of an operation not used

String Length: Bytes length != String length

Delete From Dynamic Array: 'delete' leaves a gap in array

Data Truncated: Division on int/uint values truncates the result

Data truncated: Division of integer values yields an integer value again. That means e.g.  $10 / 100 = 0$  instead of 0.1 since the result is an integer again. This does not hold for division of (only) literal values since those yield rational constants.

Pos: 164:37; 171:27; 175:6; 177:59; 180:26; 188:11; 239:51; 260:87; 275:8;  
346:148; 353:135; 353:166; 354:51; 354:82 <https://oracle-group.dev>

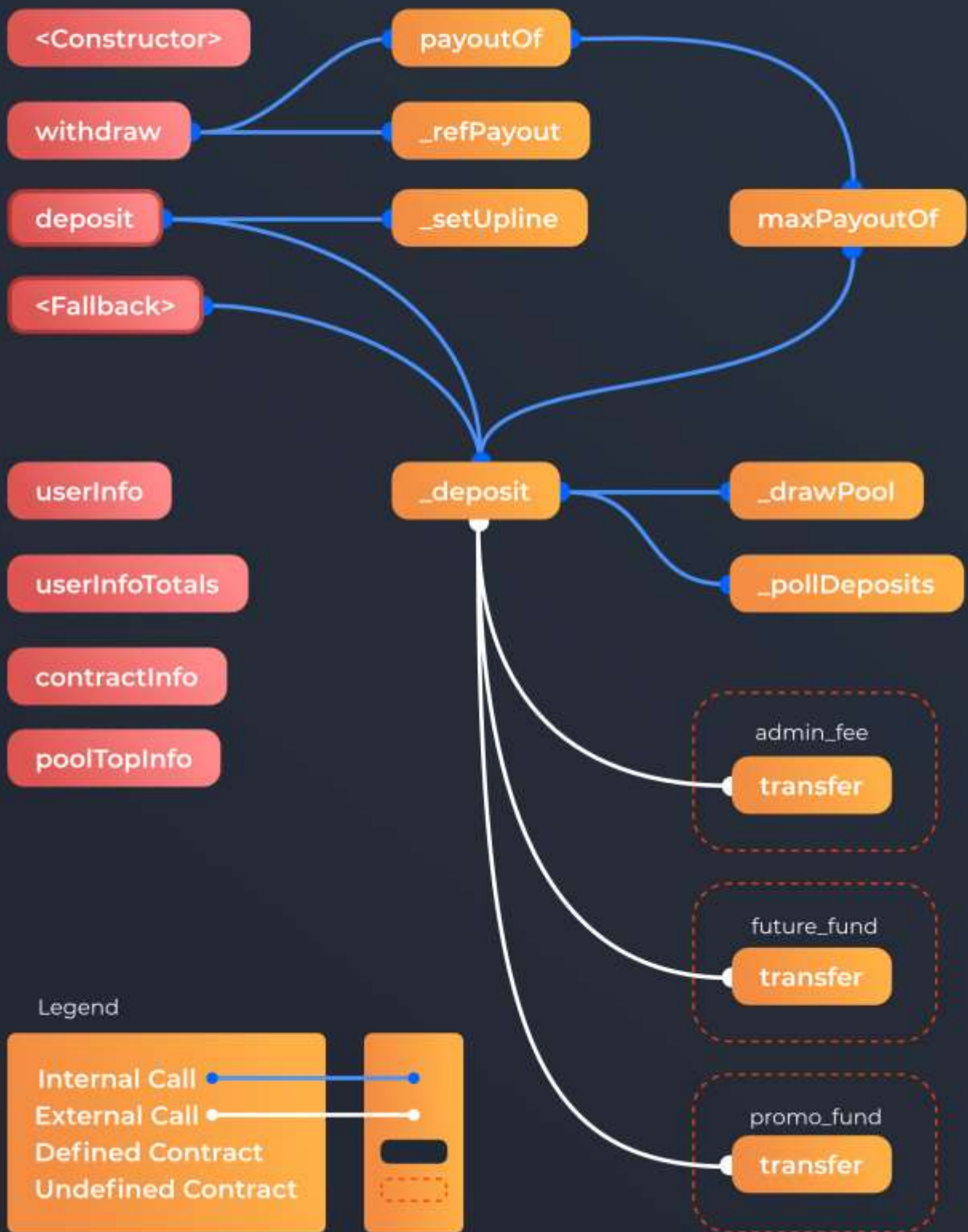
<https://github.com/OpenZeppelin/openzeppelin-contracts/blob/master/contracts/math/SafeMath.sol>

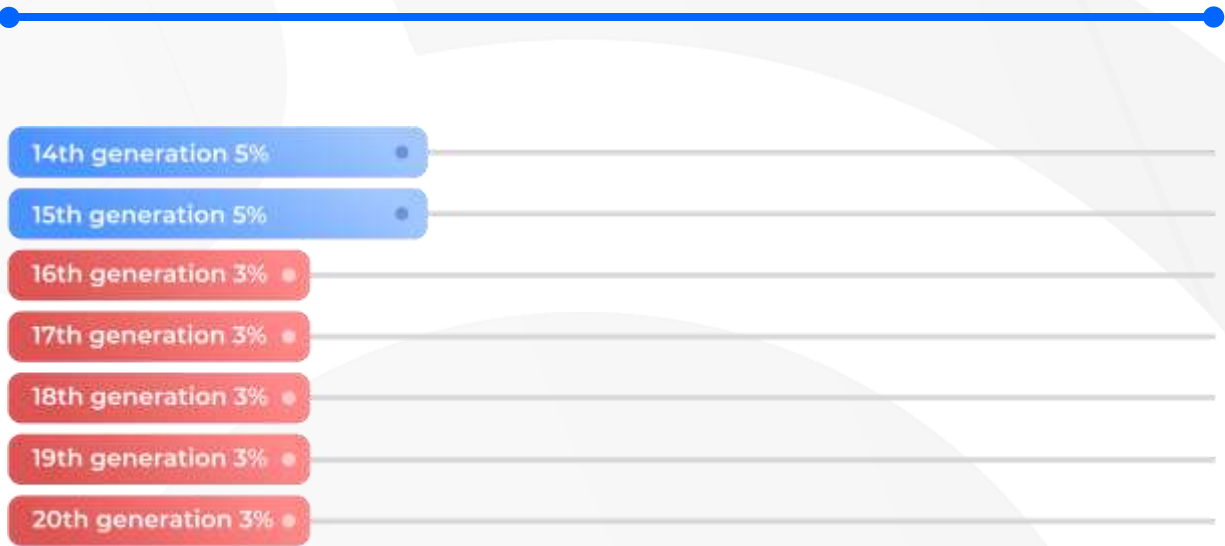
Solidity unit testing

BallotTest

Check winning proposal

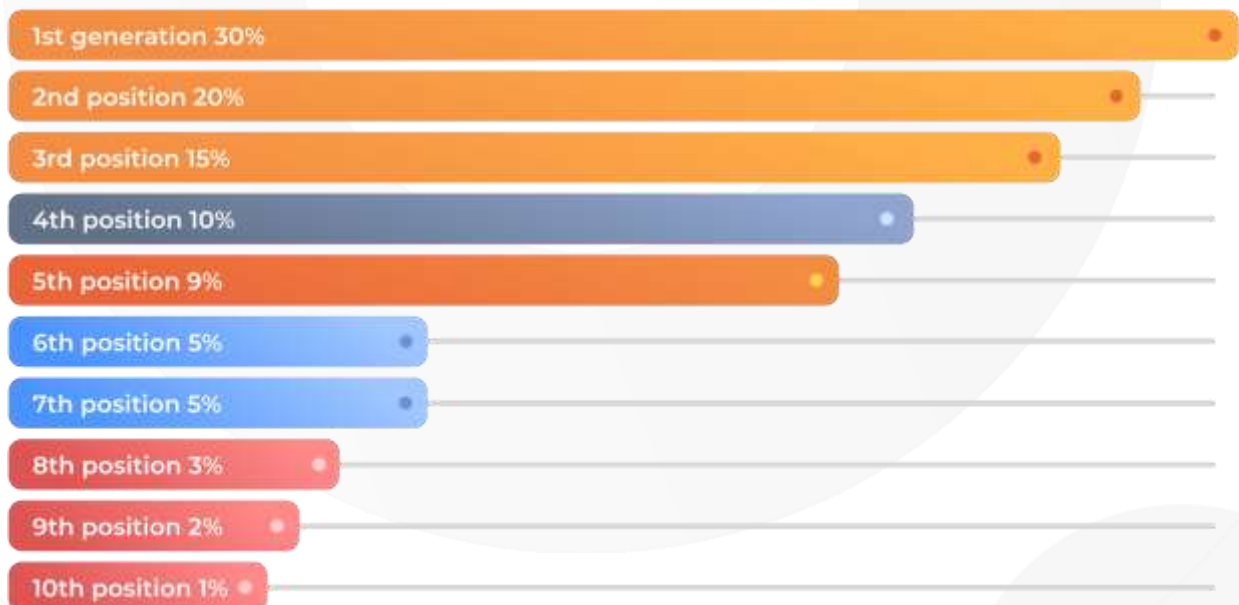
Check winning proposal with return value





1 new level is activated for each direct partner, maximum 20 levels, see above.

**ALL** Deposits set aside in pool, every 24 hour 10% of the pool is shared among top 10 sponsors in volume by



---

Minimum and maximum deposit limits

1st cycle, minimum deposit 50 TRX, up to 100 000 TRX.

2nd cycle, equal or greater than previous deposit, up to 300 000 TRX.

3rd cycle, equal or greater than previous deposit, up to 900 000 TRX.

4th cycle and beyond, equal or greater than previous deposit, up to 2 000 000 TRX.

Sending TRX directly to the contract is not supported by logic sending TRX to the contract from another contract is not supported by logic



All transactions will fail if the contract balance is less than the withdrawal amount, when the withdrawal function is called, because contract balance is not checked

Pos: 36 This parameter is not used in any strings (mapping(uint8 => address))










public permanent\_top; <https://oracle-group.dev>

Collect garbage every non-usable parameters for less gas

Backdoor for investor funds can be withdrawn by anyone. Bugs allowing to steal money from the contract were not detected

Symbol	Meaning
	Function can modify state
	Function is payable

## 2.1 Legend

Contract	Type	Bases		
L	Function name	Visibility	Mutability	Modifiers
TronX	Implementation			
L	<Constructor>	Public !		NO !
L	<Fallback>	External !		NO !
L	_setUpLine	Private		
L	_deposit	Private		
L	_pollDeposits	Private		
L	_refPayout	Private		
L	_drawPool	Private		
L	deposit	External !		NO !
L	withdraw	External !		NO !
L	maxPayoutOf	External !		NO !
L	payoutOf	External !		NO !
L	userInfo	External !		NO !
L	userInfoTotals	External !		NO !

---

L	contractInfo	External !		NO !
L	poolTopInfo	External !		NO !



---

## Disclaimer:

This audit is only to the Smart-Contract code at the specified address.

TYmZZsGRcdwnCPgGXSDytyzYHLPt7JKTL2

The audit makes no statements or warranties about the suitability or sustainability of the business model or regulatory rigme for the business model. Do take **caution** that you are doing all financial actions & transactions at your own risk, especially if you are dealing with high-risk projects / Dapps.