The TronexMax smart contract audit

Revision 1 dated 11.09.2020

Table of contents

Brief information

Project: TronexMax
Web: tronexmax.com
Compiler version: 0.5.12
Optimization: enabled
The audit date: 11.09.2020

Information

The contract code was reviewed and analyzed for vulnerabilities, logical errors, and the developers' exit scams possibility. This work was carried out concerning the project source code provided by the customer.

The audit revealed:
- Defining functions errors
- Other comments

The detected problems full list can be found below.

General conclusion

The audit revealed several errors, 3 comments, and 2 possible improvements that **do not affect the users'funds security on the contract. The exit scam clear signs - not found.** Errors are associated with function properties redundant definitions and the unnecessary input parameters definition. Comments and improvements are related to the contract non-optimized work and are recommendatory.

Telescr.in guarantees the TronexMax contract security and performance.

Liability disclaimer

The telescr.in team within this audit framework is not responsible for the developers or third parties actions on the platforms associated with this project (websites, mobile applications, and so on). The audit confirms and guarantees only the smart contract correct functioning in the revision provided by the project developers (check the revision).

Confirmed by digital signature

Aggregated data

The Contract analysis was performed using the following methods:

- Static analysis
  - Checking the code for common errors leading to the most common vulnerabilities
- Dynamic analysis
  - The Contract launching and carrying out the attacks various kinds to identify vulnerabilities
- Code Review

Received data

Recommendation	**Type**	**Priority**	**Occurrence probability**
Redundant definition	Error	Low	Low
Unused parameter	Error	Low	High
Unused parameter [2]	Error	Low	High
Overflow danger	Note	Informational	Low
Raw call (...)	Note	Informational	High
Extra cycle	Note	Informational	High
Extra cycle [2]	Note	Informational	High
Unused State-variables	Improvement	Informational	High
One-string functions	Improvement	Informational	Average

A. Errors

1. Redundant definition

The initialize function is advertised as payable, but the function does not accept funds, but only sets the contract original values instead of the designer.
Recommendation: Don't use the payable definition for functions that don't accept funds.

2. Unused parameter

The buyNode function asks for a price option that is never used.
Recommendation: Don't define unused parameters in the function.

3. Unused parameter [2]

The buyNode function asks for a value option that is never used.
Recommendation: Don't define unused parameters in the function.

4. Overflow danger

The withdraw_node_income() function does not use the SafeMath library when calculating the bonus amount. There is a chance that the withdraw_node_income () function will overflow, although it is very small.
Recommendation: Use the SafeMath for all calculations.

B. Notes

1. Raw call(...)

Inside the buyNode function, there is a tokenTransferFrom call, which within itself makes a call(...). The problem is that such a call is not recommended because of a reentrancy attack danger.
Recommendation: If call (...) is necessary, you should call it at the last moment after the state contract has changed.

2. Extra cycle

Within the add_node_income method, two for cycles are defined, the first identify values to process them in the second. You can do it in one cycle.
Recommendation: Implementing business logic in a single cycle could improve readability and there would be no need to store data from the first.

3. Extra cycle [2]

Within the investing method, two for cycles are defined, and the first and second are similar, with little different depending on whether the current user has referrer'a. You can do with one cycle by putting the appropriate check inside.
Recommendation B.2.

C. Improvements

1. Unused State-variables

The contract has State variables that are not used anywhere.
Recommendation: Remove unused State variables.

2. One-string functions

There are one-string functions in the contract that not only do not bring practical benefits but also impair the contract source code readability, creating an extra attachment.
Recommendation: use explicit calls instead of such functions.

Application. Error classification

Priority	
Informational	This question is not directly related to functionality but may be important to understand.
low	This question has nothing to do with security, but it can affect some behaviour in unexpected ways.
Average	The problem affects some functionality but does not result in an economically significant user funds loss.
high	This issue can result in the user funds loss.
Probability	
low	It is unlikely that the system is in a state in which an error could occur or could be caused by any party.
Average	This problem may likely arise or be caused by some party.
high	It is highly likely that this problem could arise or could be exploited by some parties.

Application. Digital bytecode print

The audit was carried out for the code certain version on the compiler version 0.5.12 with the optimization disabled.

To check the contract bytecode for identity to the one that was analysed during the audit, you must:

1. Get contract bytecode (in any block explorer)
2. Get SHA1 from bytecode string
3. Compare with reference in this report

```
Sha1 from bytecode (non-meta data):
                 620c3b305f0edcfe61146273b7ccb8578a1fcf12
Sha1 from bytecode (with metadata):
                 f7b957347a9b7712b6e042e7877d9427c5e3a14a
Contract address:
                    TNqKuahBdBXivTpNkZ3NeXryJbC8WyG1hk
```

Check the digital print

Application. Signature of the audit report

```
{
        "address": "0x505ade8cea4db608250e503a5e8d4cb436044d2e",
        "msg": "The audit revealed several errors, 3 comments, and 2
      possible improvements that do not affect the users‘funds security on
      the contract. The exit scam clear signs - not found. Errors are
      associated with function properties redundant definitions and the
      unnecessary input parameters definition. Comments and improvements
      are related to the contract non-optimized work and are
      recommendatory. Telescr.in guarantees the TronexMax contract
      security and performance. Sha1 bytecode (with meta):
      f7b957347a9b7712b6e042e7877d9427c5e3a14a , sha1 bytecode (no meta):
      620c3b305f0edcfe61146273b7ccb8578a1fcf12 Contract address:
      TNqKuahBdBXivTpNkZ3NeXryJbC8WyG1hk",
        "sig":
      "0xed02e65b6862c2fa02ad56b493ac814e6f6a39849e9221041a21502b8fb983ce7
      93581d8f97dbe691fd1d5f7bff6950bc3c1f375f4a8b8e50dd2d397956ccb011c",
        "version": "3",
}
```

Check the signature