October 2020

Security Audit

ArgosNEX

TVjZnBcq1wTggtwL1GuRRVECxShUbkRkJm

www.grox.solutions

CRITICAL ISSUES (critical, high severity): **0**

Bugs and vulnerabilities that enable theft of funds, lock access to funds without possibility to restore it, or lead to any other loss of funds to be transferred to any party; high priority unacceptable bugs for deployment at mainnet.

ERRORS, BUGS (medium, low severity): **0**

Bugs that can trigger a contract failure, with further recovery only possible through manual modification of the contract state or contract replacement altogether.

WARNINGS (any severity): **2**

Lack of necessary security precautions; other warnings for owners and users.

OPTIMIZATION POSSIBILITIES (very low severity): **0**

Possibilities to decrease cost of transactions and data storage of Smart-Contracts.

NOTES AND RECOMMENDATIONS (very low severity): **2**

Tips and tricks, all other issues and recommendations, as well as errors that do not affect the functionality of the Smart-Contract.

Conclusion:

In the ArgosNEX Smart-Contract were found no vulnerabilities and no backdoors. The code was manually reviewed for all commonly known and more specific vulnerabilities.

So ArgosNEX Smart-Contract is safe for use in the main network.

Note: root address (owner) has possibility to upgrade levels for any users (owner can renounce this function, see ‘Warnings’).

AUDIT RESULT:

WARNINGS

1. Owner privileges (medium severity):

Root Address (address specified at deploy of smart-contract) has possibility to upgrade level of any user via ‘inception’ function. This function can be deprecated forever (not allowed to call anymore) if function ‘deprecateInception’ is called by the Root once. To check status one can use ‘inceptionDeprecated’ view function: if it returns ‘true’ owner’s upgrade of level is not callable anymore.

2. Various cost of transactions (low severity):

Costs of transactions can be various because of different individual logic (upgrading of levels/reinvests) used depending on referrer position. Also there is a loop ‘while’ inside of ‘buyLevel’ function that increase the cost of transaction in case of search of relevant upline. In case of repeating of a loop take a mention that blockchain structures as TRON and Ethereum have defined limits of size of transactions.

NOTES AND RECOMMENDATIONS:

3. View functions (very low severity):

We recommend to add public view functions to query information from smart-contract, especially to know number of referrals (which is unavailable with simple query of ‘users’ function.

4. Unnecessary parameter (very low severity):

There is ‘_level’ parameter in the ‘buy’ function, but the purchase of the next level is possible only. So that parameter can be removed, function can be restructured and renamed to ‘buyNextLevel’ function.

Note: that is recommendation only. Doesn’t really affect the logic of smart-contract.

Independent description of smart-contract functionality:

ArgosNEX smart-contract is MLM structure that distributes funds between ‘uplines’ (referrers) and ‘downlines’ (referrals).

Structure has 15 levels: price for the first level (registration) is 100 TRX. Prices for every next level is x2 of current price (100, 200, 400 etc).

User must specify an active referrer while registration. If no referrer provided transaction is reverted.

Referrer of the user is unchangeable. Amount of possible referrals is not limited.

User gets payment only if he already got the relevant level of payment. Otherwise payment goes up in the structure until it meets the relevant user.

First two payments from referrals at the level goes to upgrade of level. All next payments user gets directly on his wallet balance.

There is no owner commission fees in the project.

Root Address (address specified at deploy of smart-contract) has possibility to upgrade level of any user via ‘inception’ function. This function can be deprecated forever (not allowed to call anymore) if function ‘deprecateInception’ is called by the Root once. To check status one can use ‘inceptionDeprecated’ view function: if it returns ‘true’ - owner’s upgrade of level is not callable anymore.

List of functions of the smart-contract:

Write contract (call of these function changes state and requires tx fee payed):

1. register - buy a first level (need to specify an active referrer address).
2. buy - buy a level (actually only next level).
3. inception - upgrade level of any user for free (only Root address access).
4. deprecateInception - remove possibility to call ‘inception’ function forever.

Read contract (call of these function doesn't require tx fee payed):

1. nextLevelCost - how much should pay user to upgrade his level.
2. inceptionDeprecated - current state of ‘inception’ function: of if it returns ‘true’ - owner’s upgrade of level is not callable anymore.
3. root - address of root user.
4. levels - prices of levels.
5. users - user info.

October 2020

Disclaimer

This audit is not a call to participate in the project and applies only to the Smart-Contract code at the specified address.

Do not forget that you are doing all financial actions at your own risk, especially if you deal with high-risk projects.

If you have any questions or are interested in developing/auditing of Smart-Contracts, please contact us and we will consult you.

Telegram: **@gafagilm**
E-mail: **info@grox.solutions**

www.grox.solutions