September 2022

Security Audit

Tetherex

TVrHddERScoUvfgAViceSQz83weZpsyjdE

https://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; critical warnings for owners, customers or investors.

ERRORS, BUGS AND WARNINGS (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; Lack of necessary security precautions; other warnings for owners and users.

OPTIMIZATION POSSIBILITIES (very low severity): **2**

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

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

Warning: do not send tokens to smart-contract address using usual transfer. One must use two-step transfer only: “approve” function inside of USDT Token and “invest” function inside of Tetherex.

AUDIT RESULT:

Optimization possibilities

1. Recording statistical parameters in the blockchain (very low severity):

List of statistical parameters that also increase the cost of transactions and increase the amount of data stored in the blockchain:

```
    uint256 public totalUsers;
    uint256 public totalInvested;
    uint256 public totalWithdrawn;
    uint256 public totalDeposits;
    uint256 public totalIncomes;
```

Recommendation: use events and log this information instead of writing it to the blockchain.

Note: this comment doesn't affect the main functionality of the smart-contract.

2. Transfers inside of referral program (very low severity):

There is a 3-level referral program and all bonuses are transferred directly to the recipients inside of ‘invest’ function. That actions (up to 3 extra token transfers) increase total transaction fees.

Recommendation: that is optimally to use ‘pull payment system’ instead, when user withdraws his available referral bonuses himself.

Notes

3. Loops over dynamic variables (very low severity):

In the `withdraw`, `getUserDividends`, `getUserAvailable`, `getUserTotalDeposits`, and `getUserTotalWithdrawn` functions, loops unrestrictedly grow as the number of deposits increases. If one creates a large number of parallel deposits from a single wallet, it can lead to an excessive increase of the transaction cost.

4. Closing the last payment (very low severity).

If the last user who leaves the project has a payout greater than the smart-contract balance, he will receive the entire available balance, but it will be recorded that the entire payout was closed.

Note: this comment is not critical, since after the smart contract balance is empty, it is unlikely that the contract will be used again. So it makes sense for last user to get at least something.

Independent description of the smart-contract functionality:

The Tetherex contract provides the opportunity to invest any amount in USDT TRC20 Token (from 50 USDT) in the contract and get a 200% return on investment, if the contract balance has enough funds for payment.

Dividends are paid from deposits of users (Ponzi scheme).

Note: USDT TRC20 Token smart contract -
https://tronscan.org/#/contract/TR7NHqjeKQxGTCi8q8ZY4pL8otSzgjLj6t

Warning: do not send tokens to smart-contract address using usual transfer. Otherwise you will lost your tokens.

One must use only two-step transfer to create deposit:

1) “approve” function inside of USDT Token (allow spend tokens to Tetherex).
2) “invest” function inside of Tetherex. The minimum amount for deposit is 50 USDT.

That is because of standard critical problem of ERC20 called «event handling». More info at https://grox.solutions/all/erc20-issues

Each subsequent Deposit is kept separately in the contract, in order to maintain the payment amount for each Deposit.

The daily percentage for user dividends starts from 3% and depends on the following factors:

- Every 20,000 USDT of total invested tokens to contract +0.1% until 23%. This Contract Bonus cannot decrease.

- Every 12 hours of non-withdrawal of dividends from the contract +0.1% (when creating new deposits, the percent keeps growing).

Maximum daily percent is not limited (3+20+HoldBonus).

All dividends are calculated at the moment of request and available for withdrawal at any time. All percent bonuses are summarized and multiplied by all period of deposit.

Withdrawal is performed by calling the “withdraw” function from the address the Deposit was made. Minimum withdrawal amount is 5 USDT.

3-level referral program: in the “invest” function, one can specify the address of the referrer.

As a result, the referrer (upline) will get direct transfer of share of the investor's Deposit according to the following table:

Referrer level	1	2	3
Percent, %	6	2	1

Requirements for the referrer: you can not specify your own wallet as a referrer, as well as a wallet that does not have at least one contribution in the smart contract.

If wrong referrer is provided, default referrer will receive all bonuses until you provide referrer**.**

Once referrer is specified he is assigned to the user without the possibility of changing. The referrer will get his percents from each subsequent Deposit.

Contract owners fee 11%: part of the invested funds is sent to 3 addresses:

(marketing address) - 5%.
(platform address) - 5%.
(admin address) - 1%.

The contract contains statistical functions that do not require sending transactions:

1. `getContractBalance` – smart contract USDT balance (with 6 decimals).
2. `getContractBalanceRate` - current contract percent (with 1 decimal, 30=3%).
3. `getUserReferrer` – the user's referrer.
4. `getUserAvailable` – total available amount to withdraw.
5. `isActive` – whether the user has active deposits.
6. `getUserAmountOfDeposits` – the number of user deposits.
7. `getUserTotalDeposits` – the sum of each deposits of the user.
8. `getUserTotalWithdrawn` – user dividend withdrawal amount.
9. `getUserReferralBonus` - total ref bonus of all levels.
10. `getUserDepositInfo` - user specified (by index) deposit info: amount, withdrawn, UNIX start time.

September 2022

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.

Warning:

Beware of fake audits.
All official info available on 3 resourses only:

Website: **www.grox.solutions**
Telegram: **www.t.me/grox_solutions (@grox_solutions)**
YouTube: **www.youtube.com/c/groxsolutions**

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

Telegram: **www.t.me/gafagilm (@gafagilm)**
E-mail: **info@grox.solutions**

https://grox.solutions