Gains Network Security review

Pashov Audit Group

Conducted by: T1M0H, immeas

December 12th 2023 - December 16th 2023

Contents

1. About Pashov Audit Group

Pashov Audit Group consists of multiple teams of some of the best smart contract security researchers in the space. Having a combined reported security vulnerabilities count of over 1000, the group strives to create the absolute very best audit journey possible - although 100% security can never be guaranteed, we do guarantee the best efforts of our experienced researchers for your blockchain protocol. Check our previous work here or reach out on Twitter @pashovkrum.

2. Disclaimer

A smart contract security review can never verify the complete absence of vulnerabilities. This is a time, resource and expertise bound effort where we try to find as many vulnerabilities as possible. We can not guarantee 100% security after the review or even if the review will find any problems with your smart contracts. Subsequent security reviews, bug bounty programs and on-chain monitoring are strongly recommended.

3. Introduction

A time-boxed security review of the **gTrade-contracts** repository was done by **Pashov Audit Group**, with a focus on the security aspects of the application's smart contracts implementation.

4. About gTrade-contracts

Copied from the first security review

The `gTrade` protocol is a leveraged trading platform developed by Gains Network. It allows for up to 150x on crypto tokens, 1000x on forex and 250x on commodities. The architecture mainly revolves around the `GNS` ERC20 token and ERC721 utility tokens, both of which provide different utility when using the platform - reduced spread, governance and others.

Continued

The protocol has implemented a new version of their staking, allowing for multiple reward tokens.

5. Risk Classification

Severity	**Impact: High**	**Impact: Medium**	**Impact: Low**
Likelihood: High	Critical	High	Medium
Likelihood: Medium	High	Medium	Low
Likelihood: Low	Medium	Low	Low

5.1. Impact

- High - leads to a significant material loss of assets in the protocol or significantly harms a group of users.
- Medium - only a small amount of funds can be lost (such as leakage of value) or a core functionality of the protocol is affected.
- Low - can lead to any kind of unexpected behavior with some of the protocol's functionalities that's not so critical.

5.2. Likelihood

- High - attack path is possible with reasonable assumptions that mimic on-chain conditions, and the cost of the attack is relatively low compared to the amount of funds that can be stolen or lost.
- Medium - only a conditionally incentivized attack vector, but still relatively likely.
- Low - has too many or too unlikely assumptions or requires a significant stake by the attacker with little or no incentive.

5.3. Action required for severity levels

- Critical - Must fix as soon as possible (if already deployed)
- High - Must fix (before deployment if not already deployed)
- Medium - Should fix
- Low - Could fix

6. Security Assessment Summary

review commit hash **-** **5f31ff9bf7de2237569c195d81bbf54f48ac0be2**

fixes review commit hash **-** **a29fb9b5a475a2f516488f0e128fb96e2e1e78d8**

Scope

The following smart contracts were in scope of the audit:

- `GNSStaking`

7. Executive Summary

Over the course of the security review, T1M0H, immeas engaged with Gains Network to review gTrade-contracts. In this period of time a total of **3** issues were uncovered.

Protocol Summary

Protocol Name	gTrade-contracts
Repository	https://github.com/GainsNetwork-org/gTrade-contracts
Date	December 12th 2023 - December 16th 2023
Protocol Type	leveraged trading protocol

Findings Count

Severity	**Amount**
Medium	1
Low	2
Total Findings	**3**

Summary of Findings

ID	Title	Severity	Status
[M-01]	User can't unstake in case he gets blacklisted by reward token USDC	Medium	Acknowledged
[L-01]	Reward tokens cannot be removed	Low	Acknowledged
[L-02]	Use of transfer/transferFrom	Low	Resolved

8. Findings

8.1. Medium Findings

[M-01] User can't unstake in case he gets blacklisted by reward token USDC

Severity

Impact: High. User can't unstake his Gns

Likelihood: Low. USDC blacklisting is not usual operation.

Description

USDC is supposed to be used as reward token, note it has blacklist functionality. Thus USDC tokens can't be transferred to/from blacklisted address.

Currently harvested token rewards are transferred in a force manner to staker on any interaction: claim vested tokens, stake, unstake, revoke vesting. Additionally, in `unstakeGns()` all the rewards are processed. Therefore if any of token transfers reverts, user will be unable to harvest and therefore unstake.

While unstake it firstly proccesses rewards for all tokens:

```
function unstakeGns(uint128 _amountGns) external {
        require(_amountGns > 0, "AMOUNT_ZERO");

        harvestDai();
@>      harvestTokens();

        ... // Main unstake logic
    }
```

And then transfers harvested amount of every reward token, including USDC. As discussed above, USDC transfer can revert:

```
function _harvestToken(address _token, uint128 _stakedGns) private {
        ... // Main harvest logic

@>      IERC20(_token).transfer(msg.sender, uint256(pendingTokens));

        emit RewardHarvested(msg.sender, _token, pendingTokens);
    }
```

Thus user who got blacklisted by USDC can't harvest all token rewards and therefore unstake. His stake will exist forever, receiving portion of rewards which will never be claimed.

The same scenario is when user tries to claim vested amount.

Recommendations

Use claim pattern: instead of transferring tokens, only increase internal balances. Something like:

```
mapping (address token => uint256 pendingBalance) pendingBalances;
```

And also introduce claim method.

8.2. Low Findings

[L-01] Reward tokens cannot be removed

The new version of `GNSStaking` allows for multiple reward tokens. The contract owner adds new reward tokens through `addRewardToken`. There is however no way to remove reward tokens.

All major interactions by a user, `stakeGns`, `unstakeGns`, and `createUnlockSchedule` loops through all reward tokens to claim rewards and update latest debts. If the amount of reward tokens grows too much it is possible this could result in out of gas for the user.

Consider adding a max number of rewards tokens or a method of removing/disabling reward tokens.

[L-02] Use of `transfer`/`transferFrom`

Not all ERC20 implementations are well behaved and revert on failure. Some simply return false and some might not return anything at all (USDT on mainnet).

Consider using OpenZeppelin `SafeERC20`s `safeTransfer` and `safeTransferFrom` to be as safe as possible for any atypical ERC20 tokens.