HACKEN

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Customer: XP Network
Date: June 6th, 2021

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Document

Name	Smart Contract Code Review and Security Analysis Report for XP Network TGE - initial audit.
Approved by	Andrew Matiukhin | CTO Hacken OU
Type	Vesting
Platform	Ethereum / Solidity
Methods	Architecture Review, Functional Testing, Computer-Aided Verification, Manual Review
Repository	https://github.com/VKint/tge
Commit	4e8ce852b40dd91a4537d6630c5ab137ab62d129(commit for initial audit) 3b1bdf2a6c43239c2d4a586a876d4ecc356db703(commit for remediation check)
Changelog	23 June 2021 - Initial Audit 05 July 2021 - Remediation Check

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Table of contents

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Introduction

Hacken OÜ (Consultant) was contracted by XP Network (Customer) to conduct a Smart Contract Code Review and Security Analysis. This report presents the findings of the security assessment of Customer's smart contract and its code review conducted on June 23rd, 2021.
Remediation check was done 6th of June 2021.

Scope

The scope of the project is smart contracts in the repository:

```
Repository: https://github.com/VKint/tge
Commit: 4e8ce852b40dd91a4537d6630c5ab137ab62d129(Initial audit commit)
        3b1bdf2a6c43239c2d4a586a876d4ecc356db703(Remediation check commit)

Files:
      contracts/Migration.sol
      contracts/PrivateTokenVesting.sol
      contracts/TokenVesting.sol
      contracts/XPNET.sol
```

We have scanned this smart contract for commonly known and more specific vulnerabilities. Here are some of the commonly known vulnerabilities that are considered:

Category	Check Item
Code review	▪ Reentrancy ▪ Ownership Takeover ▪ Timestamp Dependence ▪ Gas Limit and Loops ▪ DoS with (Unexpected) Throw ▪ DoS with Block Gas Limit ▪ Transaction-Ordering Dependence ▪ Style guide violation ▪ Costly Loop ▪ ERC20 API violation ▪ Unchecked external call ▪ Unchecked math ▪ Unsafe type inference ▪ Implicit visibility level ▪ Deployment Consistency ▪ Repository Consistency ▪ Data Consistency

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Functional review	■ Business Logics Review ■ Functionality Checks ■ Access Control & Authorization ■ Escrow manipulation ■ Token Supply manipulation ■ Assets integrity ■ User Balances manipulation ■ Data Consistency manipulation ■ Kill-Switch Mechanism ■ Operation Trails & Event Generation

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Executive Summary

According to the assessment, the Customer's smart contracts is secure[1].

Insecure	Poor secured	Secured	Well-secured

You are here

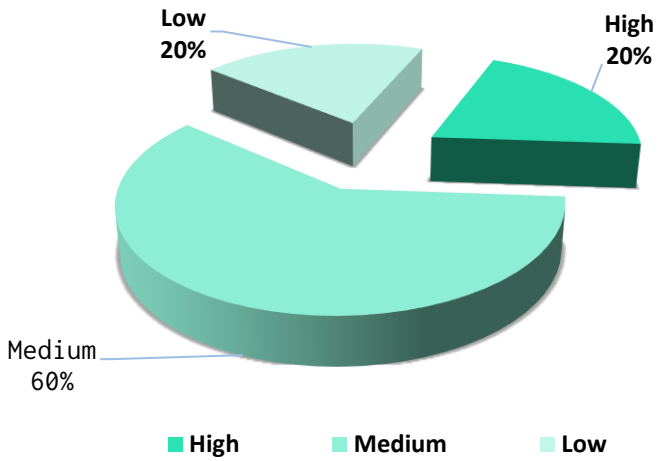
Our team performed an analysis of code functionality, manual audit, and automated checks with Mythril and Slither. All issues found during automated analysis were manually reviewed, and important vulnerabilities are presented in the Audit overview section. All found issues can be found in the Audit overview section.

As a result of the audit, security engineers 1 high, 3 medium and 1 low severity issues.

As a result of the remediation check, all issues were addressed.

Notice: test coverage of reviewed contracts is low. We recommend covering as much cases as possible.

Graph 1. The distribution of vulnerabilities after the audit.



[1] For more detail please read Audit Overview.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Severity Definitions

Risk Level	Description
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to assets loss or data manipulations.
High	High-level vulnerabilities are difficult to exploit; however, they also have a significant impact on smart contract execution, e.g., public access to crucial functions
Medium	Medium-level vulnerabilities are important to fix; however, they can't lead to assets loss or data manipulations.
Low	Low-level vulnerabilities are mostly related to outdated, unused, etc. code snippets that can't have a significant impact on execution

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Audit overview

Critical

No critical issues were found.

High

1. The contract does not guarantee that tokens will be received by beneficiaries. Owner can revoke their permissions or withdraw all tokens from the contract. Also, no tokens guaranteed when new beneficiary is created.

   Contracts: TokenVesting.sol, PrivateTokenVesting.sol

   Recommendation: forbid withdrawing all tokens and do not revoke beneficiaries; ensure that `_amount` of tokens is transferred to the contract when new beneficiary is created.

   Status: Addressed, client provided the official letter to confirm that these features are required in order to remain compatible with KYC/AML and other legal requirements. For enhanced security client confirm that the owner account shall be a multisig address.

Medium

1. `_start`, `_cliff` and `_duration` values are not validated. _vestedAmount can always fail if start is greater than cliff.

   Contracts: PrivateTokenVesting.sol

   Function: constructor

   Recommendation: validate input paramanters.

   Status: Addressed.

2. `_start` and `_cliff` parameters are not validated. This may lead to fails of the `_vestedAmount` function

   Contracts: TokenVesting.sol

   Function: createBeneficiary

   Recommendation: validate input paramanters.

   Status: Addressed.

3. `cliff` parameters is redundant and can be removed. `start` parameter can be simply increased for the corresponding value and contracts will behave the same way,

Contracts: TokenVesting.sol, PrivateTokenVesting.sol

Recommendation: remove redundant parameters.

Status: Addressed.

- Low

1. The code contains commented out fragments.

   Contracts: TokenVesting.sol, PrivateTokenVesting.sol

   Recommendation: remove commented out code.

   Status: Addressed.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Conclusion

Smart contracts within the scope were manually reviewed and analyzed with static analysis tools.

Audit report contains all found security vulnerabilities and other issues in the reviewed code.

As a result of the audit, security engineers 1 high, 3 medium and 1 low severity issues.

As a result of the remediation check, all issues were addressed.

Notice: test coverage of reviewed contracts is low. We recommend covering as much cases as possible.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Disclaimers

Hacken Disclaimer

The smart contracts given for audit have been analyzed in accordance with the best industry practices at the date of this report, in relation to cybersecurity vulnerabilities and issues in smart contract source code, the details of which are disclosed in this report (Source Code); the Source Code compilation, deployment, and functionality (performing the intended functions).

The audit makes no statements or warranties on security of the code. It also cannot be considered as a sufficient assessment regarding the utility and safety of the code, bugfree status or any other statements of the contract. While we have done our best in conducting the analysis and producing this report, it is important to note that you should not rely on this report only – we recommend proceeding with several independent audits and a public bug bounty program to ensure security of smart contracts.

Technical Disclaimer

Smart contracts are deployed and executed on blockchain platform. The platform, its programming language, and other software related to the smart contract can have its vulnerabilities that can lead to hacks. Thus, the audit can't guarantee the explicit security of the audited smart contracts.