HACKEN

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Customer: Humans Token AG
Date: December 6th, 2021

Document

Name	Smart Contract Code Review and Security Analysis Report for Humans Token AG.
Approved by	Andrew Matiukhin | CTO Hacken OU
Type	ERC20 token
Platform	Ethereum / Solidity
Methods	Architecture Review, Functional Testing, Computer-Aided Verification, Manual Review
Repository	https://github.com/0x4139/humans-smart-contract
Commit	7fee64d49359b22f4dbb848b0ab05e412f422b1c
Technical Documentation	YES
JS tests	YES
Website	humans.ai
Timeline	26 NOVEMBER 2021 – 06 DECEMBER 2021
Changelog	02 DECEMBER 2021 – INITIAL AUDIT 03 DECEMBER 2021 – SECOND REVIEW 06 DECEMBER 2021 – THIRD REVIEW

Table of contents

Introduction

Hacken OÜ (Consultant) was contracted by Humans Token AG (Customer) to conduct a Smart Contract Code Review and Security Analysis. This report presents the findings of the security assessment of the Customer's smart contract and its code review conducted between November 26th, 2021 - December 2nd, 2021.

Second review conducted on December 3rd, 2021

Third review conducted on December 6th, 2021.

Scope

The scope of the project is smart contracts in the repository:
Repository:
https://github.com/0x4139/humans-smart-contract
Commit:
7fee64d49359b22f4dbb848b0ab05e412f422b1c
Technical Documentation: Yes
- Humans (Heart Driven AI): http://humans.ai/presentation
- Tokenomics overview: http://humans.ai/tokenomics-overview
- Tokenomics detailed: http://humans.ai/tokenomics-detailed
- One pager: http://humans.ai/one-pager
- wp: http://humans.ai/litepaper

JS tests: Yes
- https://github.com/0x4139/humans-smart-contract/blob/master/test/HumansToken.js

Contracts:
lib/Author.sol
lib/DateTime.sol
lib/UnlockSchedule.sol
lib/Wallets.sol
HumansToken.sol
HumansTokenMock.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
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

Executive Summary

According to the assessment, the Customer's smart contracts are well-secured.

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 found **3** low severity issues.

After the second review security engineers found **1** medium severity issue and also wallets list was changed.

After the third review security engineers found **all issues** were addressed.

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

Audit overview

■ ■ ■ ■ Critical

No critical issues were found.

■ ■ ■ High

No high severity issues were found.

■ ■ Medium

Inconsistency in the tokenomics

While provided tokenomics document says that "Advisors & Strategic Partners" unlock should be:

- 39 000 000 Month 13
- 39 000 000 Month 19
- 58 500 000 Month 25
- 58 500 000 Month 31
- 58 500 000 Month 37
- 58 500 000 **Month 43**
- 78 000 000 Month 49

But in the code we see the following schedule:

- 39 000 000 Month 13
- 39 000 000 Month 19
- 58 500 000 Month 25
- 58 500 000 Month 31
- 58 500 000 Month 37
- 58 500 000 **Month 42**
- 78 000 000 Month 49

So the difference is that code will unlock the next amount on Month 42, while the documentation says it should be Month 43

Contracts: UnlockSchedule.sol

Recommendation: Please make sure that the schedule in the code matches the tokenomics document.

Status: Fixed

■ Low

1.   A public function that could be declared external

public functions that are never called by the contract should be declared **external** to save gas

Contracts: HumansToken.sol, Author.sol, UnlockSchedule.sol

Functions: HumansToken.initialize, HumansToken.snapshot, HumansToken.pause, HumansToken.unpause, HumansToken.TriggerTokenGenerationEvent, HumansToken.PublicSale_TGE_Unlock, HumansToken.TriggerPublicSaleScheduledUnlock, HumansToken.TriggerComunityIncentivesAndRewardsScheduledUnlock, HumansToken.TriggerPrivateSaleScheduledUnlock, HumansToken.TriggerTeamScheduledUnlock, HumansToken.Marketing_TGE_Unlock, HumansToken.TriggerMarketingScheduledUnlock, HumansToken.TriggerAdvisorsAndStrategicPartnersScheduledUnlock, HumansToken.TriggerStrategicOTCScheduledUnlock, HumansToken.LiquidityExchangeAndListings_TGE_Unlock, HumansToken.TriggerLiquidityExchangeAndListingScheduledUnlock, HumansToken.SustainableDevelopment_TGE_Unlock, HumansToken.TriggerSustainableDevelopmentScheduledUnlock, HumansToken.BusinessDevelopment_TGE_Unlock, HumansToken.TriggerBusinessDevelopmentScheduledUnlock, HumansToken.AiMiningAndStaking_TGE_Unlock, HumansToken.TriggerAIMiningAndStakingScheduledUnlock, Author.SecurityContact, UnlockSchedule.CurrentScheduleMonth

Recommendation: Use the **external** attribute for functions never called from the contract.

Status: Fixed

2.   Boolean equality

Boolean constants can be used directly and do not need to be compared to **true** or **false**.

Contracts: HumansToken.sol, UnlockSchedule.sol

Functions: HumansToken.PublicSale_TGE_Unlock, HumansToken.Marketing_TGE_Unlock, HumansToken.LiquidityExchangeAndListings_TGE_Unlock, HumansToken.SustainableDevelopment_TGE_Unlock, HumansToken.BusinessDevelopment_TGE_Unlock, HumansToken.AiMiningAndStaking_TGE_Unlock, UnlockSchedule._publicSale_scheduled_unlock, UnlockSchedule._comunity_incentives_and_rewards_scheduled_unlock, UnlockSchedule._privateSale_scheduled_unlock, UnlockSchedule._team_scheduled_unlock, UnlockSchedule._marketing_scheduled_unlock, UnlockSchedule._advisors_and_strategic_partners_scheduled_unlock, UnlockSchedule._strategic_scheduled_unlock, UnlockSchedule._liquidity_and_exchange_listings_scheduled_unlock, UnlockSchedule._sustainable_development_scheduled_unlock, UnlockSchedule._business_development_scheduled_unlock, UnlockSchedule._ai_mining_and_staking_scheduled_unlock

Recommendation: Remove the equality to the boolean constant.

Status: Fixed

3.   Too many digits

Literals with many digits are difficult to read and review.

Contracts: UnlockSchedule.sol

Functions: UnlockSchedule._initialize_tged(

Recommendation: Please consider using scientific notation and ether suffix units. For example, it is better to write: "*20e6 ether*" instead of "*2000000000000000000000000000*"

Status: Fixed

Conclusion

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

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

As a result of the audit, security engineers found **3** low severity issues.

After the second review security engineers found **1** medium severity issue and also wallets list was changed.

After the third review security engineers found **all issues** were addressed.

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 the security of the code. It also cannot be considered as a sufficient assessment regarding the utility and safety of the code, bug-free 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 the security of smart contracts.

Technical Disclaimer

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