HACKEN

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Customer: Kronos Investimentos LTDA
Date: February 10th, 2023

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 Kronos Investimentos LTDA
Approved By	Noah Jelich | Lead Solidity SC Auditor at Hacken OU
Type	ERC20 token; Staking, Voting, Bridge
Platform	EVM
Network	Ethereum, BSC
Language	Solidity
Methods	Manual Review, Automated Review, Architecture Review
Website	-
Timeline	30.08.2022 - 10.02.2023
Changelog	06.09.2022 - Initial Review 12.10.2022 - Second Review 23.11.2022 - Third Review 05.01.2023 - Fourth Review 17.01.2023 - Fifth Review 23.01.2023 - Sixth Review 10.02.2023 - Seventh Review

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 Kronos Investimentos LTDA (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 contracts.

Scope

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

Initial review scope
Repository:
https://bitbucket.org/exoworldsnft/exotoken/src/master/
https://bitbucket.org/exoworldsnft/gcredtoken/src/master/

Commit:
9e440afc910102b5f54f5f7684f19a664e7e1866
d949b98cfece019e52323eb3052940106ecb31f6
Documentation: Yes

Integration and Unit Tests: No
Deployed Contracts Addresses: No
Contracts:
File: ./contracts/BridgeBase.sol
SHA3: fe299fbee3f210dd26d7fe1d12e8237757bfbcdd5be0ef9270a8f3d44d9ead89

File: ./contracts/BridgeEth.sol
SHA3: 6626f67a7d396fb02772f373e3f214e207685b24dbeda98f45e7a1b48a2d4eff

File: ./contracts/BridgeVe.sol
SHA3: ca74f30c4736413174fbcbde305706a9cf266e3c8289ee3156d8e6390c790362

File: ./contracts/ExoToken.sol
SHA3: 435488441e8effde4b676801743fecb9bacc392f122761d9d2e203b0f39795d7

File: ./contracts/IToken.sol
SHA3: 0c0ced36ecf932d395c0ad142943637f12c7a58ebc41bd93200f5e8d14c10c4f

File: ./contracts/Migrations.sol
SHA3: 7e9eea5beb83748f819e4e24e8d8cdfbd7148931b5929d7dea4a4a6b9a286c84

File: ./contracts/ProxyAdmin.sol
SHA3: 06a03fe92a017abf71f764f2588722d15de45d29338b5307f405681f004ca885

File: ./contracts/TokenEth.sol
SHA3: 0354f63b8663fc3023aff9ded204d03f5a797ac1e306e57877ff4aa20b766a68

File: ./contracts/TokenVe.sol
SHA3: 37797563286f95aaf4967762357845f92365f4bc72aaea99c1f11f3d2f814b3a

File: ./contracts/TransparentUpgradeableProxy.sol
SHA3: ce99019794d30c1bd620cfbb6b5303ea7866b38470ac1e0b4b4d1db944a38877
File: ./contracts/BridgeBase.sol
SHA3: fe299fbee3f210dd26d7fe1d12e8237757bfbcdd5be0ef9270a8f3d44d9ead89

File: ./contracts/BridgeEth.sol
SHA3: 6626f67a7d396fb02772f373e3f214e207685b24dbeda98f45e7a1b48a2d4eff

File: ./contracts/BridgeVe.sol
SHA3: ca74f30c4736413174fbcbde305706a9cf266e3c8289ee3156d8e6390c790362

File: ./contracts/GcredToken.sol
SHA3: 0080feb8416821894fb34daf6fa9c1ffe47297486fd1c1f91426927ea1ce5cea

File: ./contracts/IToken.sol
SHA3: 0c0ced36ecf932d395c0ad142943637f12c7a58ebc41bd93200f5e8d14c10c4f

File: ./contracts/Migrations.sol
SHA3: 7e9eea5beb83748f819e4e24e8d8cdfbd7148931b5929d7dea4a4a6b9a286c84

File: ./contracts/ProxyAdmin.sol
SHA3: 06a03fe92a017abf71f764f2588722d15de45d29338b5307f405681f004ca885

File: ./contracts/TokenEth.sol
SHA3: 918b47411d13147b92ab1d2cdb2c377b6aca3254bd907300dfb945938aa40c70

File: ./contracts/TokenVe.sol
SHA3: 28ee8463a228511b868a52e9d39307e3bdcb9487c8576c9c26c146403452015d

File: ./contracts/TransparentUpgradeableProxy.sol
SHA3: ce99019794d30c1bd620cfbb6b5303ea7866b38470ac1e0b4b4d1db944a38877

Second review scope

Repository:
https://bitbucket.org/exoworldsnft/exotoken/src/

Commit:
84c582bc417533da028d2f59924676b32aea117a

Documentation: Yes

Integration and Unit Tests: Yes
Deployed Contracts Addresses: No
Contracts:

File: ./contracts/bridge/Bridge.sol
SHA3: 3c0d08bfe09e87813f25c0c764d87581a98deadb981f933d0bb698c600f4350d

File: ./contracts/governance/Governance.sol
SHA3: efb503b3438b3803f85757594ba470af7c57fb471068981be33a39894706477e

File: ./contracts/interface/IBridge.sol
SHA3: f96f45151711ad2e45d6654a099114e18786d0f5220c90adf19708f7d8d8f90a

File: ./contracts/interface/IBridgeToken.sol
SHA3: 7beb93f0cdef88cb3876ede714b26cbd33a25396374af3229dce4a6315971385

File: ./contracts/interface/IEXOToken.sol
SHA3: 5ea353211565a9bf1dac686db22f9fad47b7a6ebaa001e44177b86a2074feb6d

File: ./contracts/interface/IGCREDToken.sol
SHA3: 5a1a8c6ebb26d8ed4690caee0a58916c138fcb1f7d89a6a1a175b2e98f35c922

File: ./contracts/interface/IGovernance.sol
SHA3: bfcf926b7f70c9649f1b90b246c0aaf80a9a916f99ffb57c23114b1f0fb6b188

File: ./contracts/interface/IStakingReward.sol

SHA3: fa6b4daeafd0c23e5c275f9e2bfca57b397471e2d76cc45bce82a084abef9275

File: ./contracts/Migrations.sol
SHA3: ccbc4c1b8ff42f5c4feaa8d890e50843bc8817c5f8c870d24fdc385a5e6c48c8

File: ./contracts/ProxyAdmin.sol
SHA3: a7134c844a92819fe36b42436dd0fef1a1bad0eb74eae030f35d5fc05fc1a01c

File: ./contracts/staking/StakingReward.sol
SHA3: c7225c0c16a841c0e2a9488f60717d2f39cdf36ebe2ce4a647b9b3fe3f568988

File: ./contracts/token/EXOToken.sol
SHA3: 19fcd9bbbc48639aab4cb29beee3555dd19c8ed65681e7c4f52aab901b99e4b4

File: ./contracts/token/GCREDToken.sol
SHA3: 8a029a5b361a5a03515e997e841603f5e3c9222d3d58866fdbfd7f68551b853b

File: ./contracts/TransparentUpgradeableProxy.sol
SHA3: a7ecccf36dfa86e7737288060881d5c3652743c079411456d89dddf6c92fbaf1

Third review scope

Repository:
https://bitbucket.org/exoworldsnft/exotoken/src/

Commit:
c33b5a3b818bc0f7f969cc350035c026b1a7cacb
Documentation: Yes

Integration and Unit Tests: Yes
Deployed Contracts Addresses: No
Contracts:

File: ./contracts/bridge/Bridge.sol
SHA3: 6e380b8ced2f5c0cd23808bfb96ee3199a59bb179d4881095c772d9bb827cf85

File: ./contracts/governance/Governance.sol
SHA3: efb503b3438b3803f85757594ba470af7c57fb471068981be33a39894706477e

File: ./contracts/interface/IBridge.sol
SHA3: f96f45151711ad2e45d6654a099114e18786d0f5220c90adf19708f7d8d8f90a

File: ./contracts/interface/IBridgeToken.sol
SHA3: 7beb93f0cdef88cb3876ede714b26cbd33a25396374af3229dce4a6315971385

File: ./contracts/interface/IEXOToken.sol
SHA3: 5ea353211565a9bf1dac686db22f9fad47b7a6ebaa001e44177b86a2074feb6d

File: ./contracts/interface/IGCREDToken.sol
SHA3: 5a1a8c6ebb26d8ed4690caee0a58916c138fcb1f7d89a6a1a175b2e98f35c922

File: ./contracts/interface/IGovernance.sol
SHA3: bfcf926b7f70c9649f1b90b246c0aaf80a9a916f99ffb57c23114b1f0fb6b188

File: ./contracts/interface/IStakingReward.sol
SHA3: 1a36368a4d5d5c8d233eca1514294e8cfda1066537e11c1b7d97d81a51d2ab2b

File: ./contracts/Migrations.sol
SHA3: ccbc4c1b8ff42f5c4feaa8d890e50843bc8817c5f8c870d24fdc385a5e6c48c8

File: ./contracts/ProxyAdmin.sol

SHA3: a7134c844a92819fe36b42436dd0fef1a1bad0eb74eae030f35d5fc05fc1a01c

File: ./contracts/staking/StakingReward.sol
SHA3: fe310d06f16f923a76c61abeb6bc6d6b5dcae834906998cda531ac377d7e6074

File: ./contracts/token/EXOToken.sol
SHA3: f328a04b3b3b98c4865cb4e1aca88ba8cdc81df20f6c9990eb2a0d62f30e5d2e

File: ./contracts/token/GCREDToken.sol
SHA3: 333865312bb889ca1ae77ed5eef8d9b979d361f5f6f56f9f234b989ea9cd6158

File: ./contracts/TransparentUpgradeableProxy.sol
SHA3: a7ecccf36dfa86e7737288060881d5c3652743c079411456d89dddf6c92fbaf1

Fourth review scope

Repository:
https://bitbucket.org/exoworldsnft/exotoken/src/master/

Commit:
7b728934edfd5383762d8db2353a998dd39e2f27
Documentation: Yes

Integration and Unit Tests: Yes
Deployed Contracts Addresses: No
Contracts:

File: ./contracts/bridge/Bridge.sol
SHA3: fcd90b2fd8c868ad0b3295dc5e10c9f7b24acbaced72d92855194e31bc1fada3

File: ./contracts/governance/Governance.sol
SHA3: 81bcfd2920bb8295149448a978685c83279adddf41abe6265ac1670d28b74ec3

File: ./contracts/interface/IBridge.sol
SHA3: 561726af0474b85da34e0017399414c61f17a3b0e3d16838d7d0aa0331f184c5

File: ./contracts/interface/IBridgeToken.sol
SHA3: 7acae9c5a9ea2290fe478f37d049bdf955269e7885d7f8b5871ea7520da241a9

File: ./contracts/interface/IEXOToken.sol
SHA3: 1ffbbc0c5ad36f5f004609d9417d1798f512eb05a634a8c67b3e0ab1a8b12868

File: ./contracts/interface/IGCREDToken.sol
SHA3: 2e208155d4b2707eb5fbe0c816d10a7ff8c8a8305363428cc80670c24ff57631

File: ./contracts/interface/IGovernance.sol
SHA3: fda7c4fbb42d07065b32b95641d096fa9e19ee3e102980fa388f3eb88043d3d0

File: ./contracts/interface/IStakingReward.sol
SHA3: 85e81548fd96a74358551f58e4ca343cc5a7f0fdb283d69d9d4a13a59a50b36e

File: ./contracts/Migrations.sol
SHA3: d9803a7c3383473fc5836d50c36a9e600a646b078845847bc93c3080dca9590f

File: ./contracts/ProxyAdmin.sol
SHA3: dd365754abcf8df3e2a765f16ebf170f4af99114ff32d1926131247a5891e77b

File: ./contracts/staking/StakingReward.sol
SHA3: 3d7da28a8ff9302ee676d7fa9ed682d027d0398d887395d569d03c2b641c6a13

File: ./contracts/token/EXOToken.sol

SHA3: 471058ebd5c04fedb0ac8599476adb4efe59a3a4a84aa5b742a0f48e205a89a0

File: ./contracts/token/GCREDToken.sol
SHA3: c662257ad6a8eb491b37fec057f27c9ec1e170b0f5dc8eaf2220a54d208a18e4

File: ./contracts/TransparentUpgradeableProxy.sol
SHA3: 391658c5c79da48486ffb9b16f49520e635f26275dad27a4136f225e9b45afb5

Fifth review scope

Repository:
https://bitbucket.org/exoworldsnft/exotoken/src/master/

Commit:
c808cc58079973c0c1ffdde5f7fa3e850ffd4738
Documentation: Yes

Integration and Unit Tests: Yes
Deployed Contracts Addresses: No
Contracts:

File: ./contracts/bridge/Bridge.sol
SHA3: 60f2856476c16a947ae5d69b8bbb5b5b836a492d2fb6cd558f803facac451f2f

File: ./contracts/governance/Governance.sol
SHA3: fcfeffb8c12f37e80765af38d4745271a35f573c10b7bb66dcbae15d166c8ec7

File: ./contracts/interface/IBridge.sol
SHA3: 561726af0474b85da34e0017399414c61f17a3b0e3d16838d7d0aa0331f184c5

File: ./contracts/interface/IBridgeToken.sol
SHA3: 7acae9c5a9ea2290fe478f37d049bdf955269e7885d7f8b5871ea7520da241a9

File: ./contracts/interface/IEXOToken.sol
SHA3: 1ffbbc0c5ad36f5f004609d9417d1798f512eb05a634a8c67b3e0ab1a8b12868

File: ./contracts/interface/IGCREDToken.sol
SHA3: 3343a2569b23c41bf83dfa170a976f79d43db8dc1f239993c5fac3528cfe02c3

File: ./contracts/interface/IGovernance.sol
SHA3: 6dcae622c56cecc25dce8d8fa3f4ea0298629346f9c88887970d5d5fdc9b1238

File: ./contracts/interface/IStakingReward.sol
SHA3: bbd7fcdd9ceb8f5d34d80db0e05e80ab76a1db7755ac45f9af69d25c7baef078

File: ./contracts/Migrations.sol
SHA3: 2ffa5456fea86fc6554685944c105d258404e735a0ae490af40b526499060de6

File: ./contracts/ProxyAdmin.sol
SHA3: abeae828d5f6630be5f03eb82803a376b38ed5665a83d8be772e548ab0904681

File: ./contracts/staking/StakingReward.sol
SHA3: e6d08945585c7418d68f66dfbbac7b8ecc4fb1040b71780cbe9cc586c5458dac

File: ./contracts/token/EXOToken.sol
SHA3: 11310a3ac263c7aff0db6720d93e5e31683e5618bdb5417c29bdd3bd05dc441f

File: ./contracts/token/GCREDToken.sol
SHA3: ac7ae86f79f88eb23d83eeffc674646f9253d7b1d975bc879a66d2c9cf05240b

File: ./contracts/TransparentUpgradeableProxy.sol

SHA3: 51eb4103f5b3e26c058a9cdb83b8943dcf2c5a3e91e403d3ab6a8a07f0237884

Sixth review scope

Repository:
https://bitbucket.org/exoworldsnft/exotoken/src/master/

Commit:
df63d15910a706b68322839fc2d4d57c69f959a3
Documentation: Yes

Integration and Unit Tests: Yes
Deployed Contracts Addresses: No
Contracts:
File: ./contracts/bridge/Bridge.sol
SHA3: 9efb2fbaed6c1806dec87aeab001cb052822a7f6ca8e195af7e1b2fc82fe961c

File: ./contracts/governance/Governance.sol
SHA3: 4acc0bf03cd53f61e32719c5f6a8748fc1b4d98cdf0693cc68d49d55b4569a92

File: ./contracts/interface/IBridge.sol
SHA3: 561726af0474b85da34e0017399414c61f17a3b0e3d16838d7d0aa0331f184c5

File: ./contracts/interface/IBridgeToken.sol
SHA3: 7acae9c5a9ea2290fe478f37d049bdf955269e7885d7f8b5871ea7520da241a9

File: ./contracts/interface/IEXOToken.sol
SHA3: 1ffbbc0c5ad36f5f004609d9417d1798f512eb05a634a8c67b3e0ab1a8b12868

File: ./contracts/interface/IGCREDToken.sol
SHA3: 3343a2569b23c41bf83dfa170a976f79d43db8dc1f239993c5fac3528cfe02c3

File: ./contracts/interface/IGovernance.sol
SHA3: 6dcae622c56cecc25dce8d8fa3f4ea0298629346f9c88887970d5d5fdc9b1238

File: ./contracts/interface/IStakingReward.sol
SHA3: bbd7fcdd9ceb8f5d34d80db0e05e80ab76a1db7755ac45f9af69d25c7baef078

File: ./contracts/Migrations.sol
SHA3: 2ffa5456fea86fc6554685944c105d258404e735a0ae490af40b526499060de6

File: ./contracts/ProxyAdmin.sol
SHA3: abeae828d5f6630be5f03eb82803a376b38ed5665a83d8be772e548ab0904681

File: ./contracts/staking/StakingReward.sol
SHA3: b98b7b090ce78933ecb13e513857ded4cc082eb3ddc621471465ec9b61225852

File: ./contracts/token/EXOToken.sol
SHA3: 4806640fbd3afc74b97ea12717692d8f4de98fa080de75162fca1d03d1016d01

File: ./contracts/token/GCREDToken.sol
SHA3: e927c812dc086a6ab89259768c4b9eacbe145d47c4685be06e419fbc26c2a0d8

File: ./contracts/TransparentUpgradeableProxy.sol
SHA3: 51eb4103f5b3e26c058a9cdb83b8943dcf2c5a3e91e403d3ab6a8a07f0237884

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 cannot lead to assets loss or data manipulations.
Low	Low-level vulnerabilities are mostly related to outdated, unused, etc. code snippets that cannot have a significant impact on execution.

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

Executive Summary

The score measurement details can be found in the corresponding section of the scoring methodology.

Documentation quality

The total Documentation Quality score is **8** out of **10**.

- Tokenomics and some functional requirements were provided.
- Technical description for running the development environment is provided.
- NatSpec format was followed.
- A public whitepaper is provided but it includes some spelling and alignment mistakes.

Code quality

The total Code Quality score is **9** out of **10**.

- Code conforms to the style guide rules and does not include any redundant declaration.
- StakingReward contract has low readability due to use of lots of variables.

Test coverage

Unit tests are provided and all are running properly.
Test coverage of the project is 97.47%.

Security score

As a result of the audit, the code contains no severity issue. The security score is **10** out of **10**.

All found issues are displayed in the “Findings” section.

Summary

According to the assessment, the Customer's smart contract has the following score: **9.5**.

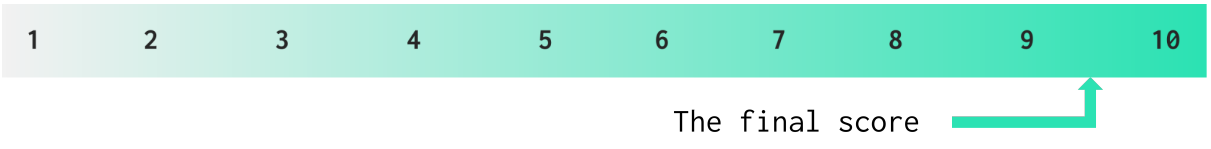

Table. The distribution of issues during the audit

Review date	Low	Medium	High	Critical
6 September 2022	**24**	**4**	**8**	**6**

12 October 2022	**10**	**1**	**2**	**4**
23 November 2022	**7**	**0**	**0**	**1**
05 January 2023	**7**	**0**	**0**	**0**
17 January 2023	**4**	**0**	**0**	**0**
23 January 2023	**0**	**0**	**1**	**0**
10 February 2023	**0**	**0**	**0**	**0**

Checked Items

We have audited provided smart contracts for commonly known and more specific vulnerabilities. Here are some of the items that are considered:

Item	Type	Description	Status
Default Visibility	SWC-100 SWC-108	Functions and state variables visibility should be set explicitly. Visibility levels should be specified consciously.	Passed
Integer Overflow and Underflow	SWC-101	If unchecked math is used, all math operations should be safe from overflows and underflows.	Passed
Outdated Compiler Version	SWC-102	It is recommended to use a recent version of the Solidity compiler.	Passed
Floating Pragma	SWC-103	Contracts should be deployed with the same compiler version and flags that they have been tested thoroughly.	Passed
Unchecked Call Return Value	SWC-104	The return value of a message call should be checked.	Passed
Access Control & Authorization	CWE-284	Ownership takeover should not be possible. All crucial functions should be protected. Users could not affect data that belongs to other users.	Passed
SELFDESTRUCT Instruction	SWC-106	The contract should not be self-destructible while it has funds belonging to users.	Not Relevant
Check-Effect-Interaction	SWC-107	Check-Effect-Interaction pattern should be followed if the code performs ANY external call.	Passed

Assert Violation	SWC-110	Properly functioning code should never reach a failing assert statement.	Passed
Deprecated Solidity Functions	SWC-111	Deprecated built-in functions should never be used.	Passed
Delegatecall to Untrusted Callee	SWC-112	Delegatecalls should only be allowed to trusted addresses.	Passed
DoS (Denial of Service)	SWC-113 SWC-128	Execution of the code should never be blocked by a specific contract state unless it is required.	Passed
Race Conditions	SWC-114	Race Conditions and Transactions Order Dependency should not be possible.	Passed
Authorization through tx.origin	SWC-115	tx.origin should not be used for authorization.	Passed
Block values as a proxy for time	SWC-116	Block numbers should not be used for time calculations.	Passed
Signature Unique Id	SWC-117 SWC-121 SWC-122 EIP-155	Signed messages should always have a unique id. A transaction hash should not be used as a unique id. Chain identifier should always be used. All parameters from the signature should be used in signer recovery	Not Relevant
Shadowing State Variable	SWC-119	State variables should not be shadowed.	Passed
Weak Sources of Randomness	SWC-120	Random values should never be generated from Chain Attributes or be predictable.	Not Relevant
Incorrect Inheritance Order	SWC-125	When inheriting multiple contracts, especially if they have identical functions, a developer should carefully specify inheritance in the correct order.	Passed
Calls Only to Trusted Addresses	EEA-Level-2 SWC-126	All external calls should be performed only to trusted addresses.	Passed
Presence of unused variables	SWC-131	The code should not contain unused variables if this is not justified by design.	Passed
EIP standards violation	EIP	EIP standards should not be violated.	Passed
Assets integrity	**Custom**	Funds are protected and cannot be withdrawn without proper permissions.	Passed
User Balances	**Custom**	Contract owners or any other third party	Failed

manipulation		should not be able to access funds belonging to users.	
Data Consistency	**Custom**	Smart contract data should be consistent all over the data flow.	Passed
Flashloan Attack	**Custom**	When working with exchange rates, they should be received from a trusted source and not be vulnerable to short-term rate changes that can be achieved by using flash loans. Oracles should be used.	Not Relevant
Token Supply manipulation	**Custom**	Tokens can be minted only according to rules specified in a whitepaper or any other documentation provided by the Customer.	Passed
Gas Limit and Loops	**Custom**	Transaction execution costs should not depend dramatically on the amount of data stored on the contract. There should not be any cases when execution fails due to the block Gas limit.	Passed
Style guide violation	**Custom**	Style guides and best practices should be followed.	Passed
Requirements Compliance	**Custom**	The code should be compliant with the requirements provided by the Customer.	Passed
Environment Consistency	**Custom**	The project should contain a configured development environment with a comprehensive description of how to compile, build and deploy the code.	Passed
Secure Oracles Usage	**Custom**	The code should have the ability to pause specific data feeds that it relies on. This should be done to protect a contract from compromised oracles.	Not Relevant
Tests Coverage	**Custom**	The code should be covered with unit tests. Test coverage should be 100%, with both negative and positive cases covered. Usage of contracts by multiple users should be tested.	Passed
Stable Imports	**Custom**	The code should not reference draft contracts, that may be changed in the future.	Passed

System Overview

Kronos Investimentos is a system that includes 2 different ERC20 tokens; EXO token, GCred token. EXO tokens can be used for staking and voting; as a staking reward, users get GCred tokens + EXO tokens. GCred tokens can be used for buying an item.

- Bridge - a simple contract that builds a bridge and allows to mint/burn tokens.
- EXOToken - an upgradable and pausable ERC20 token contract with different functionalities.
  It has the following attributes:
    - Name: ExoToken
    - Symbol: EXO
    - Decimals: 18
    - Total supply: 10 billion.
- IExoToken - interface of ExoToken contract.
- Migrations - a contract stores the number of the last deployment script applied.
- ProxyAdmin- the exact same contract of OpenZeppelin ProxyAdmin.
- TransparentUpgradeableProxy - a proxy contract with a built-in admin and upgrade interface. The contract is from OpenZeppelin library.
- GCREDToken - an upgradable ERC20 contract that allows users to buy items in exchange for GCred tokens.
    - Name: ExoToken
    - Symbol: EXO
    - Decimals: 18
    - Total supply: not capped.
- Governance - an upgradable contract that manages a voting system. Having one unit of EXO token is enough to vote for a proposal.
- StakingReward - staking contract that allows users to stake their EXO tokens. As a reward, both EXO and GCRED tokens are distributed.

Privileged roles

- The owner of the ExoToken, TokenEth, and TokenVe contract can:
    - pause/unpause the contract
    - update the bridge address
    - update the GCred token address
    - update the FNWallet address
    - mint EXO tokens
    - can distribute users' staked tokens and rewards.
    - create a new subject for voting
- The admin of the Bridge contract can:
    - mint EXO tokens
- The owner of the GCred contract can:
    - pause/unpause the contract

- update the EXO token address
- set MDWallet and DAOWallet addresses
- mint GCred tokens

Risks

- Bridge contracts are secure; however, bridge protocol is centralized, and secureness relies on the backend server. The security of the protocol cannot be guaranteed.
- The owner can burn EXO tokens from any address without permission.

Findings

■■■■ Critical

1. Token Supply Manipulation

According to the Tokenomics documentation provided by the Customer, EXO token will have a maximum supply of 10 billion tokens. In the contract, the owner or the bridge address can mint tokens infinitely.

Path: ./exotoken/contracts/ExoToken.sol

Recommendation: Cap the total supply of EXO tokens.

Status: Fixed (Revised commit: c33b5a3b818bc0f7f969cc350035c026b1a7cacb)

2. Invalid Calculations

On lines 249-260, decreasing the *tierStatus[msg.sender]* by one is done 2 times.

This causes invalid calculations, and if the *tierStatus* is “1”, the transaction always will fail since the result of 1-1-1 is not an unsigned integer.

Path: ./exotoken/contracts/ExoToken.sol: transfer()

Recommendation: Remove one of the subtracting operations.

Status: Fixed (Revised commit: 84c582bc417533da028d2f59924676b32aea117a)

3. Users Cannot Withdraw All Funds

“if” statement on line 360 has a logic that gives the reward amount if the date is expired; else, it only gives the staked amount back. However, there is no option to get both earned reward and staked amount when the reward time is up. Moreover, it is impossible to withdraw staked amount for the following scenario: if a user’s *tierStatus* is 3 and the user decides to stake(with *staking* function) and set the duration parameter as 1. The reason for that is the “if” statement on line 375.

Path: ./exotoken/contracts/ExoToken.sol: multiClaim()

Recommendation: Fix the logic issue.

Status: Fixed (Revised commit: c33b5a3b818bc0f7f969cc350035c026b1a7cacb)

4. Possible Denial of Service

Every user’s stake is recorded in the *stakingInfos* array. *claimBatch* function iterates over all the stakes in the array. When the array reaches a large enough size, function will fail due to exceeding Gas.

getStakingInfos and *_getRewardFromFN* functions may fail since they iterate over *stakingInfos* array elements.

Therefore, users may not get their staked balances and rewards.

Path: ./exotoken/contracts/ExoToken.sol: getStakerInfo(), staking(), multiClaim(), rewardFromFn()

Recommendation: Change StakingInfo[] to mapping(address=>StakingInfo[]). Implement push-over-pull patterns on _getRewardFromFN and multiClaim functions to avoid DOS conditions. This will solve stakingInfos by returning the requested address directly from the mapping, _getRewardFromFN and claimBatch functions by making them loop-free.

Status: Fixed (Revised commit: c33b5a3b818bc0f7f969cc350035c026b1a7cacb)

5. **Data Consistency**

buy_item function takes the user's GCred tokens but does not send anything as an item.

Path: ./gcred/contracts/GcredToken.sol

Recommendation: Send the user's item or fix the issue in a different way.

Status: Fixed (Revised commit: c33b5a3b818bc0f7f969cc350035c026b1a7cacb)

6. **Possible Denial of Service**

In the ExoToken, voteFlag mapping keeps only one boolean for each unique address. The users are limited to voting only once for all of the voting polls by making the addresses' boolean values turn true in the first vote, which is not reversible in the future.

This issue leads to the voting part of the contract malfunctioning.

Path: ./exotoken/contracts/ExoToken.sol: holderVote()

Recommendation: Change mapping(address => bool) to mapping(address => mapping(uint => bool)). Then, change require(!voteFlag[msg.sender]) to require(!voteFlag[msg.sender][_voteID]) and voteFlag[msg.sender] = true line to voteFlag[msg.sender][_voteID] = true. This solves the bug by giving a unique vote right to each of the users for each voting process instead of one vote right.

Status: Fixed (Revised commit: 84c582bc417533da028d2f59924676b32aea117a)

7. **Data Inconsistency**

EXO tokens are transferred from "from" address to "to" address. Therefore, both addresses need to be checked against new tier status. However, msg.sender's balance is checked for the tier status. msg.sender can not be even the from address.

This will cause assigning inaccurate tier statuses.

Path: ./exotoken/contracts/ExoToken.sol: _afterTokenTransfer

Recommendation: Check “from” and “to” addresses’ EXO token balance.

Status: Fixed (Revised commit: 7b728934edfd5383762d8db2353a998dd39e2f27)

■ ■ ■ High

1. Data Consistency

reward_from_FN variable is declared, and the *staking* function is used to assign a value. However, the variable is not used anywhere. It does not have any effect on functionality.

Path: ./exotoken/contracts/ExoToken.sol

Recommendation: Use the variable *reward_from_FN* in the intended way or remove it.

Status: Fixed (Revised commit: 84c582bc417533da028d2f59924676b32aea117a)

2. Data Consistency

During staking, users’ StakerInfo struct records *isHardStaker* and *isSoftStaker* variables into the array; these variables are never updated. Their value is always zero. So, having these variables has no effect on functionality.

Path: ./exotoken/contracts/ExoToken.sol

Recommendation: Update the variables as intended and use them in staking function or remove.

Status: Fixed (Revised commit: 84c582bc417533da028d2f59924676b32aea117a)

3. Highly Permissive Owner Access

Only the owner is allowed to distribute the rewards and staked amount. Users can withdraw neither their staked amounts nor rewards.

Path: ./contracts/staking/StakingReward.sol

Recommendation: Allows users to withdraw their staked amounts and rewards separately.

Status: Fixed (Revised commit: c33b5a3b818bc0f7f969cc350035c026b1a7cacb)

4. Highly Permissive Owner Access

The owner can burn EXO tokens from any address.

This may lead some users to lose their EXO tokens.

Paths: ./exotoken/contracts/ExoToken.sol,

./gcredtoken/contracts/GcredToken.sol

Recommendation: Do not allow the owner to burn users' holding tokens.

Status: Mitigated (Customer specified that the ecosystem and game require contract owner to be able to burn EXO tokens from any address and permissionless in case any exploit happens in the game to interfere immediately.) (Revised commit: 84c582bc417533da028d2f59924676b32aea117a)

5. Wrong Function Logic

customTransfer function is for restricting the transfer operation for users. According to the *cutomTransfer* function, users must pay some GCred tokens for each transfer. However, the *transfer* function in ERC20.sol contract is open(not overridden) and can be called by anyone.

This will cause users to transfer tokens without paying any fee.

Path: ./gcredtoken/contracts/GcredToken.sol: cutomTransfer()

Recommendation: Override the *transfer* function in GcredToken.sol.

Status: Fixed (Revised commit: 84c582bc417533da028d2f59924676b32aea117a)

6. Missing Check Against Unset State Variables

In the case of admin, MDwallet, and DAOwallet variables are being used before setting _sendGcred function before setting GCREDToken, staking function before setting FNwallet, they will cause malfunctioning accordingly.

Path: ./exotoken/contracts/ExoToken.sol: _sendGcred(), staking()

Recommendation: Check GCRED token, and FNwallet against zero address.

Status: Fixed (Revised commit: 84c582bc417533da028d2f59924676b32aea117a)

7. Unoptimized Getter Functions

ExoToken's *allStakers*, *getStakerInfo*, and *allVotes* functions try to return all the stakers, stake info, all votes, and current votes in the contract accordingly. However, in the future, as the stakers increase the *allStakers*, as the number of stakes increases the *getStakerInfo*, as the size of the total vote pool increases *allVotes*(), and as might be an error.

Path: ./exotoken/contracts/ExoToken.sol: allStakers(), getStakerInfo(), allVotes(), currentVotes()

Recommendation: Give a start and end index to the allStakers to replace the start and end value of the allStakers function and add a

requirement statement that checks whether the given and the indexed is less than the staker list or not.

Give a start and end index to the getStakerInfo function to replace the start and end value of the getStakerInfo function and add a requirement statement that checks whether the given and the indexed is less than the previous stakings' list.

Give a start and end index to the allVotes function to replace the start and end value of the allVotes function and add a requirement statement that checks whether the given and the indexed is less than the previous stakings' list.

Give a start and end index to the currentVotes function to replace the start and end value of the currentVotes function and add a requirement statement that checks whether the given and the indexed is less than the previous stakings' list.

Status: Fixed (Revised commit: 84c582bc417533da028d2f59924676b32aea117a)

8. Data Consistency

FN_REWARD variable is altered in each *stake* function execution according to the staked amount. There is no update logic, and at that moment, the amount of the last staked amount is the value.

Path: ./contracts/staking/StakingReward.sol

Recommendation: Fix the logic issue.

Status: Fixed (Revised commit: c33b5a3b818bc0f7f969cc350035c026b1a7cacb)

9. Requirements Violation

GCRED reward rates and vote weights are not correct in the provided documentation.

This may cause users to get different reward amounts than promised.

Paths: ./contracts/staking/StakingReward.sol: Lines 345-365

./contracts/governance/Governance.sol: Lines 195-198

Recommendation: Correct the values as the same as the ones in the code.

Status: Fixed (Revised commit: c808cc58079973c0c1ffdde5f7fa3e850ffd4738)

Medium

1. Unfinalized Code

There is no implementation to trigger and finalize the voting period. It only restricts the users' ability to vote for an expired proposal.

Path: ./exotoken/contracts/ExoToken.sol

Recommendation: Implement functionality that lets users know when the voting is over and see the results.

Status: Mitigated (with Customer notice) (Revised commit: 84c582bc417533da028d2f59924676b32aea117a)

2. Unfinished Code

The provided code should be implemented in the full logic of the project. isHardStaker and isSoftStaker booleans are never logically used besides the StakerInfo struct.

Path: ./exotoken/contracts/ExoToken.sol

Recommendation: Complete the code to ensure these variables are further used in the implemented functions or remove them.

Status: Fixed (Revised commit: 84c582bc417533da028d2f59924676b32aea117a)

3. Unfinished Code

In the ExoToken, stakerCnt variable increases in the staking function. StakingCnt is incremented by 1 each time the staking function is called, but it does not check whether the caller's address has already been previously added to stakerCnt or not.

In this case, stakerCnt indicates the total staking operation count, not the staker count.

Path: ./exotoken/contracts/ExoToken.sol: staking()

Recommendation: Increment stakerCnt if stakinCnt[msg.sender] is equal to zero.

Status: Fixed (Revised commit: 84c582bc417533da028d2f59924676b32aea117a)

4. Users' Tier May Decrease During Staking

In the ExoToken stake function, overridden transfer function is being used to transfer the staked balance to the contract. However, the transfer function reduces the tier of the caller if the user's remaining balance after the stake is less than the minimum staked amount.

This issue causes an unintended tier decrease.

Path: ./exotoken/contracts/ExoToken.sol: staking()

Recommendation: Fix the logic issue.

Status: Fixed (Revised commit: c33b5a3b818bc0f7f969cc350035c026b1a7cacb)

Low

1. Style Guide Violation

The provided projects do not follow the official style guidelines.

Paths: all

Recommendation: Follow the official guideline: https://docs.soliditylang.org/en/v0.8.13/style-guide.html

Status: Fixed (Revised commit: c808cc58079973c0c1ffdde5f7fa3e850ffd4738)

2. **Typos in the Code**

A typo is detected in the contract. The variable name is *unStakableAmount,* but it should be *unstakableAmount*.

Path: ./exotoken/contracts/ExoToken.sol

Recommendation: Fix the typo.

Status: Fixed (Revised commit: 84c582bc417533da028d2f59924676b32aea117a)

3. **Commented Code Parts**

On line 146, the code line is commented.

This increases the code complexity and makes the code looks unfinalized.

Path: ./exotoken/contracts/ExoToken.sol

Recommendation: Remove the commented lines.

Status: Fixed (Revised commit: 84c582bc417533da028d2f59924676b32aea117a)

4. **Redundant Variable**

claimFlag boolean variable is redundant since it is always “true” when staking is started.

Path: ./exotoken/contracts/ExoToken.sol

Recommendation: Remove the variable.

Status: Fixed (Revised commit: 84c582bc417533da028d2f59924676b32aea117a)

5. **Confusing Function Name**

Although the function returns only the sub-title name of the given VOTE session and the number of its votes, it is named as getList. It does not return any list. This may be confusing for users.

Path: ./exotoken/contracts/ExoToken.sol: getList()

Recommendation: Rename the function with a more suitable name, such as getVoteCount.

Status: Fixed (Revised commit: 84c582bc417533da028d2f59924676b32aea117a)

6. Use of OpenZeppelin Libraries from Local

ProxyAdmin and TransparentUpgradableProxy contracts are from OpenZeppelin but not imported from "@openzeppelin" library in the node module directory.

Paths: ./exotoken/contracts/ProxyAdmin.sol, ./exotoken/contracts/TransparentUpgradableProxy.sol, ./gcred/contracts/ProxyAdmin.sol, ./gcred/contracts/TransparentUpgradableProxy.sol

Recommendation: Import the contracts from OpenZeppelin library.

Status: Fixed (Revised commit: 84c582bc417533da028d2f59924676b32aea117a)

7. Typos in the Code

A typo is detected in the contract. The function name, `cutomTransfer` has a typo.

Path: ./gcred/contracts/GcredToken.sol

Recommendation: Rename the function as *customTransfer*.

Status: Fixed (Revised commit: 84c582bc417533da028d2f59924676b32aea117a)

8. Floating Pragma

Locking the pragma helps ensure that contracts do not accidentally get deployed using, for example, an outdated compiler version that might introduce bugs that affect the contract system negatively.

Paths: all

Recommendation: Consider locking the pragma version whenever possible and avoid using a floating pragma in the final deployment.

Status: Fixed (Revised commit: 84c582bc417533da028d2f59924676b32aea117a)

9. State Variables that Could Be Declared Immutable

token variable of Bridge contract is not changing after deployment, which means it is supposed to be immutable.

Path: ./exotoken/contracts/Bridge.sol

Recommendation: Declare mentioned variable immutable.

Status: Fixed (Revised commit: df63d15910a706b68322839fc2d4d57c69f959a3)

10. Missing Zero Address Validation

Address parameters are being used without checking against the possibility of 0x0.

This can lead to unwanted external calls to 0x0.

Paths: ./exotoken/contracts/BridgeBase.sol: constructor(), mint()

./gcred/contracts/BridgeBase.sol: constructor(), mint()

./exotoken/contracts/ExoToken.sol: bridgeUpdateAdmin(), changeGCRED(), changeFNwallet(),

Recommendation: Implement zero address checks.

Status: Fixed (Revised commit: df63d15910a706b68322839fc2d4d57c69f959a3)

11. Boolean Equality

Boolean constants can be used directly and do not need to be compared to true or false. In *x* functions of the *y* boolean constants used to be compared to true or false.

Paths: ./exotoken/contracts/BridgeBase.sol: mint()

./gcred/contracts/BridgeBase.sol: mint()

Recommendation: Remove boolean equality.

Status: Fixed (Revised commit: 84c582bc417533da028d2f59924676b32aea117a)

12. Redundant Use of SafeMath

Since Solidity v0.8.0, the overflow/underflow check is implemented via ABIEncoderV2 on the language level - it adds the validation to the bytecode during compilation.

There is no need to use the SafeMath library.

Paths: ./exotoken/contracts/ExoToken.sol

./gcredtoken/contracts/GcredToken.sol

Recommendation: Remove the SafeMath library.

Status: Fixed (Revised commit: 84c582bc417533da028d2f59924676b32aea117a)

13. Unused Variable

The variables GCRED, interest, and _list_ are never used.

Path: ./exotoken/contracts/ExoToken.sol

Recommendation: Remove unused variables.

Status: Fixed (Revised commit: 84c582bc417533da028d2f59924676b32aea117a)

14. Missing Event Emitting

Events for critical state changes should be emitted for tracking things off-chain.

Path: ./exotoken/contracts/ExoToken.sol: bridgeUpdateAdmin(), changeGCRED(), changeFNwallet()

Recommendation: Create and emit related events.

Status: Fixed (Revised commit: 84c582bc417533da028d2f59924676b32aea117a)

15. State Variables' Default Visibility

Variable`s MAX_REWRAD, CLAIM_DELAY of StakinReward, decimal, _totalSupply of ExoToken are not specified. Specifying state variables' visibility helps to catch incorrect assumptions about who can access the variable.

Paths: ./contracts/staking/StakinReward.sol

./contracts/token/ExoToken.sol

Recommendation: Specify variables as public, internal, or private. Explicitly define visibility for all state variables.

Status: Fixed (Revised commit: 84c582bc417533da028d2f59924676b32aea117a)

16. Stake Duration Can Not Be Zero

stakePeriod list holds 4 different variables, three of which are stored as timestamps due to the minutes data type. However, the first index of the list, 0, can not be stored as staking time can not be equal to zero.

Path: ./exotoken/contracts/ExoToken.sol

Recommendation: Declare the above-mentioned variables as constants.

Status: Fixed (Revised commit: c33b5a3b818bc0f7f969cc350035c026b1a7cacb)

17. Variable Shadowing

In the transfer function of ExoToken, the owner variable shadows an existing variable.

In the transfer function of GcredToken, the owner variable shadows an existing variable.

Paths: ./exotoken/contracts/ExoToken.sol

./gcredtoken/contracts/GcredToken.sol

Recommendation: Rename related variables/arguments.

Status: Fixed (Revised commit: 84c582bc417533da028d2f59924676b32aea117a)

18. Unoptimized Getter Function

Gas usage of getList function is unoptimized.

Path: ./exotoken/contracts/ExoToken.sol: getList()

Recommendation: Get _voteID from mapVote and save it to a temporary memory variable. Then, return its title and vote count.

Vote memory tmp_vote = mapVote[_voteID]

return tmp_vote[_voteId].title, tmp_vote[_voteId].voteCnt

Status: Fixed (Revised commit: 84c582bc417533da028d2f59924676b32aea117a)

19. The Redundant for Loop

There are two for loops in futureVotes function. The first one iterates all the previous votes and checks whether their start dates are passed or not. The second for loop iterates allVotes and marks not started votes as future votes.

Path: ./exotoken/contracts/ExoToken.sol: futureVotes()

Recommendation: Instead of the second iteration, it is more Gas efficient to store the required indexes during the first iteration, and iterate over those indexes instead of the whole list.

Status: Fixed (Revised commit: 84c582bc417533da028d2f59924676b32aea117a)

20. Functions that Can Be Declared as External

In order to save Gas, public functions that are never called in the contract should be declared as external.

Paths: ./exotoken/contracts/token/EXOToken.sol: pause(), unpause()

./exotoken/contracts/bridge/Bridge.sol: pause(), unpause()

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

Status: Fixed (Revised commit: 84c582bc417533da028d2f59924676b32aea117a)

21. Function Name - Functionality Mismatch

The function's name buy_item and its functionality does not match. There is no buying process in the function, as the function name suggests. The function divides the given amount and transfers it to MDwallet and DAOwallet.

Path: ./gcredtoken/contracts/GcredToken.sol: buy_item()

Recommendation: Rename the function.

Status: Fixed (Revised commit: c33b5a3b818bc0f7f969cc350035c026b1a7cacb)

22. Confusing Contract Names

TokenVe and TokenEth contract names are confusing since their initialize function creates the token with the name of ExoToken, the same as the ExoToken contract.

Paths: ./gcredtoken/contracts/TokenEth.sol

./gcredtoken/contracts/TokenVe.sol

Recommendation: Rename the contracts and clarify the chains used.

Status: Fixed (Revised commit: 84c582bc417533da028d2f59924676b32aea117a)

23. Typos in the Code

A typo is detected in the contract. The event name is `UnStake` is mistaken.

Path: ./contracts/staking/StakingReward.sol

Recommendation: Fix the typo as “Unstake”.

Status: Fixed (Revised commit: c33b5a3b818bc0f7f969cc350035c026b1a7cacb)

24. Redundant Function Declaration

`getTierMinAmount` and `_getTierMinAmount` functions are doing the same thing. To save Gas and to increase code readability, do not declare redundant functions.

Path: ./contracts/staking/StakingReward.sol

Recommendation: Remove the _getTierMinAmount function.

Status: Fixed (Revised commit: c33b5a3b818bc0f7f969cc350035c026b1a7cacb)

25. Typos in the Code

A typo was found in the contract. The constant name `MAX_REWRAD` is wrong.

Path: ./contracts/staking/StakingReward.sol

Recommendation: Fix the typo as “MAX_REWARD”.

Status: Fixed (Revised commit: 84c582bc417533da028d2f59924676b32aea117a)

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 by the best industry practices at the date of this report, with 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 report contains no statements or warranties on the identification of all vulnerabilities and security of the code. The report covers the code submitted to and reviewed, so it may not be relevant after any modifications. Do not consider this report as a final and sufficient assessment regarding the utility and safety of the code, bug-free status, or any other contract statements.

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.

English is the original language of the report. The Consultant is not responsible for the correctness of the translated versions.

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, Consultant cannot guarantee the explicit security of the audited smart contracts.