AUDITONE

Audit Report

AURORA

Created on: 14.03.2023

Audit Report

by AuditOne

Smart Contract Security Analysis Report

Note: This report may contain sensitive information on potential vulnerabilities and exploitation methods. This must be referred internally and should be only made available to the public after issues are resolved (to be confirmed prior by the client and AuditOne).

Table of Contents

Introduction

Csanuragjain, Defsec and Blocksec team, who are auditors at AuditOne, successfully audited the smart contracts (as indicated below) of Aurora. The audit has been performed using manual analysis. This report presents all the findings regarding the audit performed on the customer's smart contracts. The report outlines how potential security risks are evaluated. Recommendations on quality assurance and security standards are provided in the report.

Project Description

The Aurora environment consists of the Aurora Engine, a high performance EVM—Ethereum Virtual Machine—and the Rainbow Bridge, facilitating trustless transfer of ETH and ERC-20 tokens between Ethereum and Aurora, within a great user experience.

Aurora exists and is operated as an independent, self-funded initiative, but will continue to leverage the shared team DNA and continually evolving technology of the NEAR Protocol.

The governance of Aurora will take a hybrid form of a Decentralized Autonomous Organization—the AuroraDAO—complemented by a traditional entity which will hold one of several seats in the AuroraDAO.

This audit focused on the NEAR plugins, which are implementations of common patterns used for NEAR smart contracts.

Project and Audit Information

Term	Description
Auditor	Csanuragjain, Defsec and Blocksec team
Reviewed by	Chinedum
Type	NEAR Plugins
Language	Rust
Ecosystem	Near protocol
Methods	Manual Review
Repository	https://github.com/aurora-is-near/near-plugins
Commit hash (at audit start)	c4db1654ffeb8736c009d967c5486523306c265a
Commit hash (after resolution)	8920f9a72631f1b224ed93405640bc062f2a0191
Documentation	[Added once the whitepaper is published by the project]
Unit Testing	No
Website	https://aurora.dev/
Submission Time	2022-12-06
Finishing Time	2023-03-14

Contracts in scope

- *near-plugins/src/access_control_role.rs*
- *near-plugins/src/access_controllable.rs*
- *near-plugins/src/events.rs*
- *near-plugins/src/full_access_key_fallback.rs*
- *near-plugins/src/lib.rs*
- *near-plugins/src/ownable.rs*
- *near-plugins/src/pausable.rs*
- *near-plugins/src/test_utils.rs*
- *near-plugins/src/upgradable.rs*
- *near-plugins-derive/src/access_control_role.rs*
- *near-plugins-derive/src/access_controllable.rs*
- *near-plugins-derive/src/full_access_key_fallback.rs*
- *near-plugins-derive/src/lib.rs*
- *near-plugins-derive/src/ownable.rs*
- *near-plugins-derive/src/pausable.rs*
- *near-plugins-derive/src/utils.rs*
- *near-plugins-derive/src/upgradable.rs*
- *Testing with NearSDK (https://github.com/near/near-sdk-rs) to verify compliance with expected behaviour*
- *Analysing pull requests where plugin is imported in the repos - (https://github.com/aurora-is-near/rainbow-bridge, https://github.com/aurora-is-near/rainbow-token-connector, https://github.com/aurora-is-near/near-erc20-connector).*

Executive Summary

Aurora plugin's smart contracts were audited between 2022-12-06 and 2023-03-14 by Csanuragjain, Defsec and Blocksec audit. Manual analysis was carried out on the code base provided by the client. The following findings were reported to the client. For more details, refer to the findings section of the report.

S.no.	Issue Category	Issues found	Resolved	Acknowledged
1.	High	2	2	0
2.	Medium	4	4	0
3.	Low	3	3	0
4.	Quality Assurance	8	8	0

Severity definitions

Risk factor matrix	Low	Medium	High
Occasional	L	M	H
Probable	L	M	H
Frequent	M	H	H

High: Funds or control of the contracts might be compromised directly. Data could be manipulated. We recommend fixing high issues with priority as they can lead to severe losses.

Medium: The impact of medium issues is less critical than high, but still probable with considerable damage. The protocol or its availability could be impacted, or leak value with a hypothetical attack path with stated assumptions.

Low: Low issues impose a small risk on the project. Although the impact is not estimated to be significant, we recommend fixing them on a long-term horizon. Assets are not at risk: state handling, function incorrect as to spec, issues with comments.

Quality Assurance: Informational and Optimization - Depending on the chain, performance issues can lead to slower execution or higher gas fees. For example, code style, clarity, syntax, versioning, off-chain monitoring (events etc.)

Audit Overview

Code quality: 95.0

Documentation quality: 98.0

Security score: 100.0

Architecture quality: Not in scope

Audit Findings

Finding: #1

Issue: Full access key is not removed on ownership change

Severity: High

Where:

https://github.com/aurora-is-near/near-plugins/blob/master/near-plugins-derive/src/full_access_key_fallback.rs#L26

Impact: If owner is changed, the public_key which was set by old owner as full access key is not removed. This allows old owner to still have access to misuse the access key say, redeploy contract

Description: The full access key is not removed on changing owner. This allow old owner to still have ownership using the access key set by him.

- Owner A sets a new full access key using attach_full_access_key function
- Owner is changed to Owner B
- Owner A still has control over the contract using the full access key (since the public key from step 1 is not removed, owner A still can use the access key to perform critical operations)

Recommendations:

- Tie access key with the owner who has set it. In case owner has changed that access key becomes useless
- Remove the access key on changing owner

Status: Resolved

Finding: #2

Issue: Mismatched Variable Names

Severity: High

Where:

https://github.com/aurora-is-near/near-plugins/blob/c4db1654ffeb8736c009d967c5486523306c265a/near-plugins-derive/src/pausable.rs#L153

https://github.com/aurora-is-near/near-plugins/blob/c4db1654ffeb8736c009d967c5486523306c265anear-plugins-derive/src/pausable.rs#L200

Impact: The related check is not implemented correctly and cannot be compiled successsfully.

Description: if users want to add except() arguments on function decrease_1 in examples/pausable-examples/pausable_base/src/lib.rs line 49-52 like:

```
    #[if_paused(name = "increase_1", except(owner, self))]
    pub fn decrease_1(&mut self) {
        self.counter -= 1;
    }
```

However, the rust compiler cannot compile it due to the error:

```
error[E0425]: cannot find value `__check_paused` in this scope
  --> pausable-examples/pausable_base/src/lib.rs:49:5
   |
49 |     #[if_paused(name = "increase_1", except(owner, self))]
   |     ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^ help: a local variable with a similar name e
   |
   = note: this error originates in the attribute macro `if_paused` (in Nightly builds, run with -Z macro-ba

For more information about this error, try `rustc --explain E0425`.
```

Recommendations: In the function get_bypass_condition, we use variable __check_paused. But in function if_paused we use variable check_paused.They should be the same variable. Change the variable check_paused to __check_paused in function if_paused

Status: Resolved.

Finding: #3

Issue: Timelock behind up_deploy_code

Severity: Medium

Where:

https://github.com/aurora-is-near/near-plugins/blob/c4db1654ffeb8736c009d967c5486523306c265a/near-plugins-derive/src/upgradable.rs#L52

Impact: a malicious owner can simply deploy malicious code instantly which could lead to stealing user funds.

Description: User should get considerable time to opt out of an upgraded code. Since up_deploy_code does not require any delay, a malicious owner can simply deploy malicious code instantly.

- An innocent contract A is deployed with high staking APR
- User starts investing using the contract
- Owner stages and deploys a malicious code which adds a new function to drain user funds
- User funds get stolen

Recommendations: Place up_deploy_code function behind timelock which gives user suitable time to make decision

Status: Resolved

Finding: #4

Issue: Lack of State Migration and State Integrality Check

Severity: Medium

Where:
https://github.com/aurora-is-near/near-plugins/blob/c4db1654ffeb8736c009d967c5486523306c265a/near-plugins-derive/src/upgradable.rs

Impact: After users deploy the update, the contract may fail to deserialize its state, which leads to a DoS problem.

Description: In the current implementation, there is no state migration interface. For example, a user updates the contract like following,

```
#[near_bindgen]
#[derive(Ownable, Upgradable, Default, BorshSerialize, BorshDeserialize)]
struct Counter {
  counter: u64,
}
```

to

```
#[near_bindgen]
#[derive(Ownable, Upgradable, Default, BorshSerialize, BorshDeserialize)]
struct Counter {
  counter: u64,
  counter2: u64,
}
```

In this case, the contract cannot deserialize the account's state due to the newly added attribute counter2.

Recommendations: To fix the problem, users need to implement a method that migrates the old state, add the counter2 to the Counter. In order to check if the migration is correct, it's highly recommended to invoke a view function to get the whole contract state after the migration is completed. Add a migration interface for user

Status: Resolved

Finding: #5

Issue: Improper Full Access Key Fallback Design

Severity: Medium

Where:

https://github.com/aurora-is-near/near-plugins/blob/c4db1654ffeb8736c009d967c5486523306c265a/near-plugins/src/full_access_key_fallback.rs

Impact: The owner of the contract has more privileges than expected.

Description: A Full Access Key could have total control over the account. It can transfer NEAR, deploy a smart contract and so on. Full access keys have more privileges than the owner and can do more things than the owner. In the current design, the owner of the contract can attach a new full-access key and can gain more privileges than expected.

Status: Resolved

Finding: #6

Issue: Pausable contract management should be handled by AccessControl plugin instead of the owner.

Severity: Medium

Where:

https://github.com/aurora-is-near/near-plugins/blob/c4db1654ffeb8736c009d967c5486523306c265a/near-plugins-derive/src/pausable.rs#L45

https://github.com/aurora-is-near/rainbow-token-connector/blob/c4db1654ffeb8736c009d967c5486523306c265a/bridge-token-factory/src/lib.rs#L224

Impact:

- Potential security vulnerabilities due to the lack of multiple points of control over the Pausable plugin.
- Lack of accountability and transparency in the management of the contract.

Description: In the current implementation, only the owner of the Pausable contract has the ability to pause or unpause it. This creates a potential security vulnerability, as the contract cannot be paused or unpaused if the owner's account is compromised or otherwise unavailable. It also lacks accountability and transparency, as it may be difficult for other stakeholders to understand why certain actions were taken and whether they were justified.

Recommendations:

- Update the Pausable contract to be managed by the AccessControl plugin instead of the owner.
- Implement additional security measures to protect the AccessControl plugin and ensure its availability.
- Ensure that all actions taken with the Pausable contract are properly documented and transparent to all stakeholders.

Status: Resolved

Finding: #7

Issue: No way to remove access key

Severity: Low

Where:

https://github.com/aurora-is-near/near-plugins/blob/c4db1654ffeb8736c009d967c5486523306c265a/near-plugins-derive/src/full_access_key_fallback.rs#L26

Impact: If private key for access key is leaked then plugin provides no way to remove the access key thus putting the contract in vulnerable state

Description: Plugin has no way to remove an added access key

Recommendations: Add a new plugin method (owner only) which allows to remove an access key for given account

Status: Resolved

Finding: #8

Issue: Unchecked Removal of the Paused_Key

Severity: Low

Where:

https://github.com/aurora-is-near/near-plugins/blob/c4db1654ffeb8736c009d967c5486523306c265a/near-plugins-derive/src/pausable.rs#L66

Impact: Users may accidentally pass the wrong Key and are not prompted.

Description: The function pa_unpause_feature() lacks a check on key's removal. The existence of the key is not checked. In this case, if the key does not exist, the contract will not panic, which may mislead the project manager and bring unexpected impacts.

Recommendations: Check the return value and panic if it is false.

Status: Resolved

Finding: #9

Issue: Lack of Macro Attributes for Pausable Derivation

Severity: Low

Where:

https://github.com/aurora-is-near/near-plugins/blob/c4db1654ffeb8736c009d967c5486523306c265a/near-plugins-derive/src/lib.rs#L31

Impact: Users cannot change the storage key of Pausable.

Description: For example, if users want to change the storage key of Pausable in examples/pausable-examples/pausable_base/src/lib.rs line 7-12,

he will add #[pausable(paused_storage_key="new_storage_key")] like:

```
#[proc_macro_derive(Pausable, attributes(pausable))]
pub fn derive_pausable(input: TokenStream) -> TokenStream {
    pausable::derive_pausable(input)
}
```

However, the rust compiler cannot compile it due to the error:

```
error: cannot find attribute `pausable` in this scope
 --> pausable-examples/pausable_base/src/lib.rs:9:3
  |
9 | #[pausable(paused_storage_key="new_storage_key")]
  |   ^^^^^^^^

error: could not compile `pausable_base` due to previous error
```

Recommendations: Add attributes(pausable) to derive_pausable.

Status: Resolved

Finding: #10

Issue: Staging code is not removed

Severity: Quality Assurance

Where:

https://github.com/aurora-is-near/near-plugins/blob/c4db1654ffeb8736c009d967c5486523306c265a/near-plugins-derive/src/upgradable.rs#L52

Impact: The upstage code is not removed on deployment which unnecessarily take storage

Description: upstage code is not removed on deployment

Recommendations: Remove upstage code on deployment

Status: Resolved

Finding: #11

Issue: Unnecessary Clone

Severity: Quality Assurance

Where:

https://github.com/aurora-is-near/near-plugins/blob/c4db1654ffeb8736c009d967c5486523306c265a/near-plugins-derive/src/ownable.rs#l20

https://github.com/aurora-is-near/near-plugins/blob/c4db1654ffeb8736c009d967c5486523306c265a/near-plugins-derive/src/pausable.rs#L28

https://github.com/aurora-is-near/near-plugins/blob/c4db1654ffeb8736c009d967c5486523306c265a/near-plugins-derive/src/upgradable.rs#L27

Impact: Redundant clones may need additional gas.

Description: The functions return Vec<u8> so we need to copy the slice on the memory to construct a new Vec.

while 'static [u8] is a reference, there is no copy needed.

Thus, we can return 'static [u8] directly.

Recommendations: Change return type from Vec<u8> to 'static [u8] and remove to_vec().

Status: Resolved

Finding: #12

Issue: Question about the design purpose of the Full Access Key Fallback

Severity: Quality Assurance

Where:

https://github.com/aurora-is-near/near-plugins/blob/c4db1654ffeb8736c009d967c5486523306c265a/near-plugins/src/full_access_key_fallback.rs

Impact: Extra consumption of gas

Description: The comment says Smart contracts can be considered trustless, when there is no Full Access Key (FAK) attached to it. Otherwise owner of the FAK can redeploy or use the funds stored on the smart contract.

However, according to the documentation, removing the contracts' FAK is suggested to make the contract fully de-centralized.

Status: Resolved

Finding: #13

Issue: Missing documentation for admins being able to remove other admins in ACL

Severity: Quality Assurance

Where:
https://github.com/aurora-is-near/near-plugins/blob/c4db1654ffeb8736c009d967c5486523306c265a/near-plugins-derive/src/access_controllable.rs#L236

Description: In current design, the admin of can revoke other admin in the same group.

Status: Resolved

Finding: #14

Issue: Replace default panic with near-sdk panic

Severity: Quality Assurance

Where:
https://github.com/aurora-is-near/near-plugins/blob/c4db1654ffeb8736c009d967c5486523306c265a/near-plugins-derive/src/ownable.rs#L115

Impact: Using the default panic function can lead to unstable behavior and make it difficult to properly handle and recover from exceptional cases. This can ultimately impact the reliability and usability of the application.

Description:
Currently, the codebase uses the default panic function to handle exceptional cases. This can lead to unexpected behavior and make it difficult to debug issues

Recommendations: To improve the reliability and stability of the application, it is recommended to replace the default panic function with the panic function provided by the near-sdk. This panic function allows for more control over the handling of exceptional cases and can make it easier to debug and recover from issues.
Implementing this change will require updating all instances of the default panic function with the near_sdk::panic function. It may also require updating any code that handles recovery from exceptional cases.

Status: Resolved

Finding: #15

Issue: revoke_super_admin_unchecked not exposed through near bindgen

Severity: Quality Assurance

Where:
https://github.com/aurora-is-near/near-plugins/blob/c4db1654ffeb8736c009d967c5486523306c265a/near-plugins-derive/src/access_controllable.rs#L236

Impact: Not exposing the revoke_super_admin_unchecked function through the near bindgen can prevent it from being used to revoke super administrative privileges as intended. This can potentially lead to security vulnerabilities if super administrative privileges are not properly managed.

Description: It has been reported that the revoke_super_admin_unchecked function is not exposed through the near bindgen, preventing it from being accessed from external interfaces.

Recommendations: To ensure that the revoke_super_admin_unchecked function can be used as intended, it is recommended to expose it through the near bindgen. This will allow the function to be accessed from external interfaces and used to revoke super administrative privileges as needed. It may also be helpful to implement safeguards to prevent accidental or malicious revocation of super administrative privileges.

Status: Resolved

Finding: #16

Issue: Open to-dos

Severity: Quality Assurance

Where:

https://github.com/aurora-is-near/near-plugins/blob/c4db1654ffeb8736c009d967c5486523306c265a//near-plugins/tests/contracts/README.md

https://github.com/aurora-is-near/near-plugins/blob/c4db1654ffeb8736c009d967c5486523306c265a/near-plugins/tests/access_controllable.rs#L79,#L375,#L382

https://github.com/aurora-is-near/near-plugins/blob/c4db1654ffeb8736c009d967c5486523306c265a/near-plugins/tests/common/utils.rs#L48

https://github.com/aurora-is-near/near-plugins/blob/c4db1654ffeb8736c009d967c5486523306c265a/near-plugins/src/upgradable.rs#L84

https://github.com/aurora-is-near/near-plugins/blob/c4db1654ffeb8736c009d967c5486523306c265a/near-plugins/src/full_access_key_fallback.rs#L46

https://github.com/aurora-is-near/near-plugins/blob/c4db1654ffeb8736c009d967c5486523306c265a/near-plugins-derive/src/ownable.rs#L93

https://github.com/aurora-is-near/near-plugins/blob/c4db1654ffeb8736c009d967c5486523306c265a/near-plugins-derive/src/access_controllable.rs#L115#L566

https://github.com/aurora-is-near/near-plugins/blob/c4db1654ffeb8736c009d967c5486523306c265a/near-plugins-derive/src/full_access_key_fallback.rs#L18

https://github.com/aurora-is-near/near-plugins/blob/c4db1654ffeb8736c009d967c5486523306c265a/near-plugins-derive/src/utils.rs#L8

Impact: The use of TODO comments in code can potentially lead to security issues if they are not properly managed and addressed. TODO comments are often used to indicate areas of code that need to be addressed or improved, but if they are not properly tracked and resolved, they can easily be overlooked and left unaddressed. This can result in unfinished or incomplete code, which can have security vulnerabilities or other issues.

Description: Open To-dos can point to architecture or programming issues that still need to be resolved. Often these kinds of comments indicate areas of complexity or confusion for developers. This provides value and insight to an attacker who aims to cause damage to the protocol.

Recommendations: Consider resolving the To-Dos before deploying code to a production context. Use an independent issue tracker or other project management software to track development tasks.

Status: Resolved

Finding: #17

Issue: Missing documentation for the case of using Upgradable to patch a vulnerability

Severity: Quality Assurance

Where:

https://github.com/aurora-is-near/near-plugins/blob/c4db1654ffeb8736c009d967c5486523306c265a/near-plugins-derive/src/upgradable.rs#L42

Impact: Attacker can see the new code in up_stage before actual deployment. Using it, attacker can determine the vulnerability in current contract code and exploit it.

Description: Before deployment all code is kept in up_stage_code. If deployment is made for security issue then up_stage_code will contain fix for vulnerable code which could be seen by users to exploit current codebase.

- User reports a critical vulnerability in contract code
- Owner quickly make patch and calls up_stage_code to write the updated code
- Before owner could deploy, attacker checks this upstage code, figures out the vulnerability and exploits it

Recommendations: For critical upgrades, encrypted code could be upstaged which could be decrypted at deployment time using a provided key.

Status: Resolved

Disclaimer

The smart contracts provided to AuditOne 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 ethical nature of the project is not guaranteed by a technical audit of the smart contract. Any owner-controlled functions should be carried out by the responsible owner. Before participating in the project, all investors/users are recommended to conduct due research.

The focus of our assessment was limited to the code parts associated with the items defined in the scope. We draw attention to the fact that due to inherent limitations in any software development process and product, an inherent risk exists that even major failures or malfunctions can remain undetected. Further uncertainties exist in any software product or application used during the development, which cannot be free from any errors or failures. These preconditions can impact the system's code and/or functions and/or operation. We did not assess the underlying third-party infrastructure, which adds further inherent risks as we rely on correctly executing the included third-party technology stack itself. Report readers should also consider that over the life cycle of any software product, changes to the product itself or the environment in which it is operated can have an impact leading to operational behaviors other than initially determined in the business specification.