LAOS Audit

August 2024

By CoinFabrik

v202311

Executive Summary

CoinFabrik was asked to audit the runtime and pallets for the LAOS project.

The LAOS project develops a polkadot parachain intended to be used as mutable storage for NFTs transacted in other blockchains.

During this audit we found no critical issues, one high-severity issue, one medium issue and several minor issues.

Scope

The audited files are from the git repository located at https://github.com/freeverseio/laos.git. The audit was made in two phases.

Phase one was conducted on commit `a3b20156f993ebf14ca214bda14b74886a79d391` and included the following directories:

- `pallets/parachain-staking`: Pallet forked from https://github.com/moonbeam-foundation/moonbeam/tree/v0.36.0/pallets/parachain-staking
- `runtime/laos`: Blockchain runtime.

Phase two was conducted on commit `16bd9e56d20c206cf927d961c8ec58f299308473` and added the following directories:

- `pallets/asset-metadata-extender`: Precompile used to associate universal locations with URIs.
- `pallets/laos-evolution`: Precompiles used to manage collections of URIs.

No other files in this repository were audited. Its dependencies are assumed to work according to their documentation. In particular it must be noted that the blockchain node was outside the scope of this audit. Also, no tests were reviewed for this audit.

Methodology

CoinFabrik was provided with the source code, including automated tests that define the expected behavior, and general documentation about the project. Our auditors spent eleven weeks auditing the source code provided, which includes understanding the context of use, analyzing the boundaries of the expected behavior of each component and function, understanding the implementation by the development team (including dependencies beyond the scope to be audited) and identifying possible situations in which the code

allows the caller to reach a state that exposes some vulnerability. Without being limited to them, the audit process included the following analyses.

- Arithmetic errors
- Race conditions
- Misuse of block timestamps
- Reentrancy attacks
- Denial of service attacks
- Excessive gas usage
- Missing or misused function qualifiers
- Needlessly complex code and code interactions
- Poor or nonexistent error handling
- Insufficient validation of the input parameters
- Incorrect handling of cryptographic signatures
- Centralization and upgradeability

Findings

In the following table we summarize the security issues we found in this audit. The severity classification criteria and the status meaning are explained below. This table does not include the enhancements we suggest to implement, which are described in a specific section after the security issues.

ID	Title	Severity	Status
HI-01	Unrestricted URI	High	Acknowledged
ME-01	Lack of URI Validation	Medium	Unresolved
MI-01	No Logging on Minting Error	Minor	Unresolved
MI-02	Lack of Universal Location Validation	Minor	Unresolved

Severity Classification

Security risks are classified as follows:

- **Critical:** These are issues that we manage to exploit. They compromise the system seriously. Blocking bugs are also included in this category. They must be fixed **immediately**.
- **High:** These refer to a vulnerability that, if exploited, could have a substantial impact, but requires a more extensive setup or effort compared to critical issues. These pose a significant risk and **demand immediate attention**.
- **Medium:** These are potentially exploitable issues. Even though we did not manage to exploit them or their impact is not clear, they might represent a security risk in the near future. We suggest fixing them **as soon as possible**.
- **Minor:** These issues represent problems that are relatively small or difficult to take advantage of, but might be exploited in combination with other issues. These kinds of issues do not block deployments in production environments. They should be taken into account and be fixed **when possible**.

Issues Status

An issue detected by this audit has one of the following statuses:

- **Unresolved**: The issue has not been resolved.
- **Acknowledged**: The issue remains in the code, but is a result of an intentional decision. The reported risk is accepted by the development team.
- **Resolved**: Adjusted program implementation to eliminate the risk.
- **Partially resolved**: Adjusted program implementation to eliminate part of the risk. The other part remains in the code, but is a result of an intentional decision.
- **Mitigated**: Implemented actions to minimize the impact or likelihood of the risk.

Critical Severity Issues

No issues found.

High Severity Issues

HI-01 Unrestricted URI

Found on commit: `16bd9e56d20c206cf927d961c8ec58f299308473`
Location:

- `pallets/laos-evolution`

Classification:
- CWE-1269: Product Released in Non-Release Configuration[1]

If a collection created via the `EvolutionCollectionFactory` precompile is set to allow public minting, any user of the LAOS blockchain may create new collection items corresponding to any slot not being used, pointing to any malicious URI, including URIs pointing to:

1. Browser exploits
2. Client-side attacks
3. Illegal images
4. Gross images
5. URIs that would only appear in the item after an evolution.

These may appear in the NFT related to the collection, and shown in the websites where NFTs are traded.

The severity of this issue was lowered given that public minting is not enabled by default.

Recommendation
Either whitelist the URIs (and owners) of the minted items when the minting is not made by the owner or disallow minting by a non-owner account.

If the second option is implemented an intermediate contract may be set as the owner and handle both the access and the allowed URIs to mint and or evolve items.

Status
Acknowledged. The development team informed us that they are aware of this issue but do not want to remove it yet, as it allows them to test bridgeless minting.

Medium Severity Issues

ME-01 Lack of URI Validation

Found on commit: `16bd9e56d20c206cf927d961c8ec58f299308473`
Location:
- `pallets/asset-metadata-extender`
- `pallets/laos-evolution`

Classification:
- CWE-1286: Improper Validation of Syntactic Correctness of Input[2]

[1] https://cwe.mitre.org/data/definitions/1269.html
[2] https://cwe.mitre.org/data/definitions/1286.html

The URIs received in the functions of the `AssetMetadataExtender` precompile and the `EvolutionCollection` precompile are not being validated nor canonized. These URIs are then stored and can be consulted from the state of the blockchain by other systems.

This may lead to exploiting these other systems with malformed URIs.

This issue makes the HI-01 Unrestricted URI issue more dangerous, as any account may write malformed URIs to be read by other systems.

Recommendation
In order to solve this issue, we recommend the string parameter receiving the URI parameter to be parsed to check that it conforms to the URI format.

In order to mitigate this issue, you may choose to document in the `.sol` files that document the interfaces of the precompiles that the format of the URI is not being validated on-chain.

Status
Unresolved.

Minor Severity Issues

MI-01 No Logging on Minting Error

Found on commit: `16bd9e56d20c206cf927d961c8ec58f299308473`
Location:

- `pallets/parachain-staking/src/rewards/mint_rewards.rs: 50-55`

Classification:

- CWE-778: Insufficient Logging[3]

If funds are not deposited for the collator when making a new block, the error is not registered anywhere.

This can be seen in the following code, where there is no else clause

```
if let Ok(amount_transferred) = T::Currency::deposit_into_existing(&collator_id, amt) {
    Self::deposit_event(Event::Rewarded {
        account: collator_id.clone(),
        rewards: amount_transferred.peek(),
    });
}
```

Recommendation
Either generate an event showing the error or at least log the error.

[3] https://cwe.mitre.org/data/definitions/778.html

Status
Unresolved.

MI-02 Lack of Universal Location Validation

Found on commit: `16bd9e56d20c206cf927d961c8ec58f299308473`
Location:

- `pallets/asset-metadata-extender`

Classification:

- CWE-1286: Improper Validation of Syntactic Correctness of Input[4]

When extending or updating an universal location with an URI using the `extendULWithExternalURI` or `updateExtendedULWithExternalURI` functions the string where the universal location is received is not validated. This string is then reflected in the emitted events that can be processed by other systems. This may lead to exploiting these other systems with malformed universal locations.

This issue has less severity than the ME-01 Lack of URI Validation issue given that the universal location is reflected only in events.

Recommendation
In order to solve this issue, we recommend the string parameter receiving the universal location parameter to be parsed to check that it conforms to the URI format.

In order to mitigate this issue, you may choose to document in the `.sol` files that document the interfaces of the precompiles that the format of the universal location is not being validated on-chain.

Status
Unresolved.

Other Considerations

The considerations stated in this section are not right or wrong. We do not suggest any action to fix them. But we consider that they may be of interest to other stakeholders of the project, including users of the blockchain, token holders or project investors.

[4] https://cwe.mitre.org/data/definitions/1286.html

These considerations correspond to the second phase of the audit, based on commit `16bd9e56d20c206cf927d961c8ec58f299308473.`

Centralization

Runtime

The runtime is configured to be administered by the `sudoer` account. A full analysis of the available functionality has not been made because a big part of the available functionality is in pallets implemented by upstream projects and outside the scope of this audit, but it must be noted that the root account has the ability to upgrade the runtime to a newer version, so it must be trusted.

The development team informed us that the network is governed by a pure proxy owned by a multisig handling the sudoer account.

EvolutionCollection Precompile

The owner of the collection can create and mutate its items. See the Privileged Roles section for details.

Upgrades

As explained in the Centralization section, the runtime can be upgraded by the sudoer account.

Privileged Roles

These are the privileged roles that we identified on the audited precompiles.

EvolutionCollection Precompile

Owner

The owner of the collection can:

1. create a new collection item via the `mintWithExternalUri` function.
2. change the URI of a collection item via the `evolveWithExternalURI` function.
3. enable new item creation by any account via the `enable_public_minting` function[5].
4. disable new item creation by any account via the `disable_public_minting` function.

[5] See the HI-01 Arbitrary URI issue for the problems that enabling public minting may lead to.

The initial owner of the collection is taken as a parameter in the `EvolutionCollectionFactory.createCollection` call that creates the collection.

Collator Payments

The monetary base in the LAOS blockchain is non-inflationary. The rewards are separated as a part of the base, and it is expected to last 2 years if no changes are made by the governance of the blockchain. If the funds allocated to the rewards are exhausted then the collators will not be paid, as it can be seen in the `mint_collator_reward` function defined in `pallets/parachain-staking/src/rewards/mint_rewards.rs:45` where errors are ignored when depositing the collator rewards[6].

Changelog

- 2024-08-09 – Initial report based on commits `a3b20156f993ebf14ca214bda14b74886a79d391` and `16bd9e56d20c206cf927d961c8ec58f299308473`.

Disclaimer: This audit report is not a security warranty, investment advice, or an approval of the LAOS project since CoinFabrik has not reviewed its platform. Moreover, it does not provide a code faultlessness guarantee.

[6] See MI-01 No Logging on Minting Error.