OpenZeppelin runtime configuration template review

Hacking assessment report

V1.0, December 16th, 2024

Haroon Basheer	haroon@srlabs.de
Louis Merlin	louis@srlabs.de
Regina Biro	regina@srlabs.de

Abstract. This work describes the result of a thorough and independent security assurance audit performed by Security Research Labs on the OpenZeppelin runtime templates. Security Research Labs is a consulting firm that has been providing specialized audit services for Substrate-based blockchains in Polkadot ecosystem since 2019.

During this audit, OpenZeppelin provided access to relevant documentation and the runtime repositories. The codebase of the runtime templates was verified to ensure that the product is resilient to hacking and abuse.

The research team identified one info-level issue regarding the deployment of custom pallets in the current runtime construct design.

Security Research Labs recommends enhancing the runtime parameter documentation by defining allowed value ranges, revisiting the design approach for runtime construction using pallet categorization, and reducing reliance on pallet macros maintained in a separate upstream repository. The current approach hinders runtime customization, limits understanding of security aspects in the Polkadot runtime configuration and abstracts critical pallet settings to default values that may not be secure for individual parachain use cases.

Content

1 Disclaimer

This report describes the findings and core conclusions derived from the audit carried out by Security Research Labs within the agreed-on timeframe and scope as detailed in Chapter 2. Please note that this report does not guarantee that all existing security vulnerabilities were exhaustively discovered in the codebase and that following all the evolution suggestions described in Chapter 3 may not ensure all future code to be bug free.

2 Motivation and Scope

OpenZeppelin [1] seeks to streamline entry into the Polkadot parachain ecosystem for developers. Their runtime configuration template offers a robust framework and an out-of-the-box solution, enabling developers to efficiently configure polkadot-sdk runtimes for both generic and EVM-compatible parachains.

An independent assessment into the runtime configuration and constructions was conducted by the Security Research Labs (SRLabs) auditors, leveraging their expertise in security auditing for various parachain projects and supporting the secure development of Polkadot-SDK libraries. The OpenZeppelin runtimes were assessed for potential security flaws, realistic attack scenarios and adherence to security best practice in the Polkadot ecosystem.

Security Research Labs previously conducted two security audits on both OpenZeppelin runtime templates as summarized in Table 1. In this follow-up engagement, the auditors focused on finding security vulnerabilities stemming from the new runtime construct design based on pallet macros and categorization in both EVM and generic runtime construction libraries.

Engagement	**Scope**	**Date**	**Reference**
1st audit	Generic template	Mar 2024	[2]
2nd audit	Generic, EVM template	Jul 2024	[3]
3rd audit	Generic, EVM template and pallet abstraction	Dec 2024	templates [4] abstractions [5]

Table 1 Runtime template audit scope

During the assessment of the runtime templates, security issues were communicated to the OpenZeppelin and Parity Security team through a dedicated Element channel.

3 Evolution suggestions

OpenZeppelin's current solution for constructing a runtime with FRAME pallets grouped into a macro-based categorization creates generic boilerplate code for polkadot-sdk developers. While this solution reflects OpenZeppelin's current use cases, this approach oversimplifies the depth of knowledge required to build a secure blockchain and limits customization options for parachain business logic. Moving the pallet configuration logic to a separate upstream repository defeats the original purpose of configurability of blockchains developed within the polkadot-sdk framework: runtime developers relying on the OpenZeppelin templates must fork two repositories to build a single custom runtime.

To enable business logic customization and enhance runtime security, it is recommended to move the pallet abstractions and templates into a mono repository format as implemented in the previous versions that have been audited. All configuration items should be documented with security implications and safe value ranges (where applicable) to optimally support customization needs and secure development.

To ensure that the runtime template with pallet macro approach is secure against known and yet undiscovered threats alike, the auditors recommend considering the evolution suggestions and best practices described in this section.

3.1 Refactor pallet configurations for better runtime security

In the EVM and Generic templates, six categories of pallets macros have all the runtime configurable type parameters defined within a single trait implementation [6]. For example, the XCM trait definition has configuration types for 7 FRAME pallets. This approach hinders development efforts when values for any of these pallets' runtime parameters need to be customized, as they are grouped into a single pallet abstraction category. As a result, the current design imposes a risk of potential misconfiguration of closely related parameters with similar variable names.

Create categories for every pallet configuration through inline comments to guide developers for setting runtime parameters efficiently when customization is required for individual pallets.

3.2 Update the documentation to reflect the pallet categories.

With the current macro approach for pallet configuration through categorization, the documentation [7] for pallet specification is not updated to reflect the pallet abstraction design principle. This creates a discrepancy from the macro-based implementation for the release tag 3.0.0-rc.

Update the documentation to reflect the current design of the runtime template and organize each pallet specification under their respective categories.

3.3 Provide parameter configuration range.

Generic and EVM runtime templates are meant to help with bootstrapping new runtime developers to accelerate development. Hence, it is critical to reason about and document pallet configurations and their values, especially when configuring parameters that have security relevance. This will ensure developers gain a deeper understanding of the inner workings of the Polkadot-SDK and effectively navigate its complexities through the runtime template.

We recommend expanding the existing documentation within the codebase to include detailed explanations of configuration parameters, the accepted range of values and the potential security pitfalls of associated misconfigurations.

3.4 Limit pallet macro usage at the runtime construct

A parachain using the OpenZeppelin runtime template will have its configuration parameters abstracted into the macro-based pallet categories. A novice runtime developer may not completely grasp the implication of code obscured by macros and deploy a custom macro without understanding its security implications to their runtime. As the runtime template is targeted towards beginners, advanced rust syntax such as macros should be used sparingly in the runtime templates.

We recommend striking a balance with creating a simplistic boilerplate runtime construct code utilizing a macro-based approach, that encourages developers to configure their project runtime construct through the type-safety system offered by Rust.

4 Findings

4.1 Summary

During the analysis of the OpenZeppelin runtime templates, Security Research Labs identified 1 info-severity issue, which is summarized in Table 1.

Issue	Template	Severity	Status
Abstractions categories do not consider custom pallets	EVM/ Generic	Info	Fix in progress

Table 2 Code audit issue summary

4.2 Abstractions categories do not consider custom pallets

Attack scenario	Deploying custom pallets in the runtime is not considered in the current pallet abstraction categories
Location	openzeppelin-pallet-abstractions/src
Attack impact	Limitations in configuration options for custom pallets in the runtime templates severely hinder usability and adoption of the OZ libraries
Severity	Info
Status	Fix in progress [8]

In the current runtime template design, pallets are categorized into 6 categories for abstraction and modular configuration using Rust macros. Pallets are broadly abstracted into XCM, System, Consensus, EVM, Assets and Governance. Each of these abstractions have subsets of FRAME pallets within them to minimize the number of lines of code in the runtime construct and provide a unified interface for configuring multiple pallets.

As parachain runtime developers may deploy additional custom pallets depending on their business logic, configuration of such pallets does not fit within the currently defined pallet abstractions. This will affect the usability of the current OZ runtime templates for projects with customized pallets, thereby hindering large scale deployment.

It is recommended to consider relevant placeholders in the pallet abstraction model for custom pallets and provide documentation for deploying these custom pallets in the runtime template.

The issue was acknowledged by the OpenZeppelin team, documentation for adding custom pallets is in progress.

5 Bibliography

[1] [Online]. Available: https://github.com/OpenZeppelin/polkadot-runtime-template/tree/c9a2c76a9db2e114eecaeba07195f9c2bdfaa094.

[2] [Online]. Available: https://github.com/OpenZeppelin/polkadot-runtime-templates/blob/v3.0.0-rc2/audits/2024-04.pdf.

[3] [Online]. Available: https://github.com/OpenZeppelin/polkadot-runtime-templates/blob/v3.0.0-rc2/audits/2024-07.pdf.

[4] [Online]. Available: https://github.com/OpenZeppelin/polkadot-runtime-templates/tree/v3.0.0-rc2 .

[5] [Online]. Available: https://github.com/OpenZeppelin/openzeppelin-pallet-abstractions/tree/v0.1-rc2.

[6] [Online]. Available: https://github.com/OpenZeppelin/polkadot-runtime-templates/blob/f50de6b7b8768a269714e11e56fcaecdbefba825/generic-template/runtime/src/configs/mod.rs#L131C1-L173C2.

[7] [Online]. Available: https://docs.openzeppelin.com/substrate-runtimes/3.0.0-rc/.

[8] [Online]. Available: https://github.com/OpenZeppelin/polkadot-runtime-templates/issues/378.