

Developer tools for Plutus Script Re-Executor

Problem Statements and Proposal Benefits

Problem Statement

Debugging live Plutus script execution on the Cardano blockchain is difficult for several reasons: the deployed on-chain scripts are typically highly optimised to reduce on-chain execution costs and they are executed in the ledger state, which is hard to construct by hand and ephemeral in the cardano-node. There is a demand for tools which assist DApp developers to be able to debug the values computed within a script against the ledger state on-chain and generate test values for scripts.

This proposal would deliver a set of extensions to the Plutus Script Re-Executor (PSR) — a tool for monitoring on-chain scripts that does not maintain its own copy of the ledger state but uses the “leashed” cardano-node, ensuring all transactions are observed while the ledger state is still accessible from the node. Such extensions would improve the debugging infrastructure of the Cardano tools lowering the resources requirement for developers (such as RAM), create new debugging scenarios using the saved scripts environments (same settings and arguments aka the correct ScriptContext as it was executed on-chain), and increase PSR’s eco-system integration.

Proposal Benefit

This proposal would enhance the already existing Plutus Script Re-Executor created for monitoring purposes of the on chain deployed scripts. It would provide a builtin web UI frontend to simplify monitoring live DApps’ execution, and gaining more information from shadow scripts as they are observed from the cardano-node. The UI would also allow for more flexible management of shadow scripts; currently shadow scripts are configured though a config file at application start but the UI would allow users to dynamically add new target and shadow scripts.

We also propose advanced usage of a storage layer to the PSR, which would retain the history of all on-chain executions of target scripts, including their ScriptContext, and the results of all shadow script executions - a lightweight, DApp specific index. By storing this information, users are able to run new scripts against previous script inputs, in a context which would no longer be accessible from a live cardano-node. This history enhances testing of DApps during development, to ensure that updates to an app either maintain previous behaviour, or fix unintended behaviour.

Proposal Details

Proposal Description

The Plutus Script Re-Executor (PSR) is a tool designed to assist the developers of DApps by providing greater visibility of Plutus script executions on the Cardano blockchain. It does this by following a *cardano-node* as it fetches blocks, detects transactions which execute a user defined set of *target* scripts, and allows the user to specify substitute *shadow* scripts to be executed in the same script context. This allows DApp developers to substitute alternative versions of their scripts, for example, with trace statements compiled in, to gain visibility to intermediate values that would usually be impossible to see due to on-chain cost saving optimisations or too expensive to keep the copy of the ledger state.

During the development of the PSR, several extensions which would improve the experience of Cardano DApp operators and developers were identified. These include:

- A web frontend UI - the current PSR is implemented as a simple REST/websocket service, providing events and the ability to query execution events. To make use of the tool, another service needs to interact with it. A web app would improve monitoring a live DApp's execution, and gaining more information from shadow scripts as they are observed from the cardano-node.
 1. The UI should be reactive and update dynamically. It should display live metrics such as: currently selected block, number of failed executions, number of cancelled executions, etc.
 2. It should also contain a list of execution events and a status (unknown, cancelled, committed) associated with each event. When the executed event is clicked, detailed information about the event should be displayed. This can include execution units, traces etc.
 3. The execution list will be paginated and is gleaned from the database. The status of all the events in the current page will be polled and refreshed every few seconds.
- Management of substitute scripts at runtime - allowing the user to add new shadow scripts after program launch. This would facilitate developers working to remove bugs or improve performance by allowing them to test new versions of a live script against all previously known script inputs.

The UI would use this API for runtime management of the set of scripts. These are currently configured in a config file at application start. The UI would let users add target script hashes, for example when developers release a new version of an app, and to add and remove shadow scripts associated with a target script.

- Script execution against the saved ScriptContexts - the PSR has the ability to store the script context for every execution of a target script on-chain; providing a way for user to change the already saved ScriptContext and re-run scripts would allow developers to

test their scripts easily, changing the on-chain environment that might be helpful in debugging.

- Export of saved data - the generation of test inputs for cardano transactions is also quite an involved process; tools like Tweag's *cooked-validators* help developers generate realistic transaction inputs for property based testing and auditing purposes. However, using the inputs from actual on-chain executions would provide a useful set of unit tests to ensure either the absence of unintended changes to a script, or validate that bugs which existed in previous versions had been fixed. This proposal would add tools to export saved on-chain events data, including ScriptContexts which can be used as golden tests for scripts in any language.
- Ecosystem integration - such things as Docker containers, configuration files for easier deployment, and documentation can boost the integration of PSR in the community.

Dependencies

- Plutus Script Re-Executor
- Modifications to ouroboros-consensus etc. implementing leashing (part of the original PSR proposal).

Maintenance

- The re-executor would need an update in advance of hard forks, since each alters the codec for blocks and may alter the execution context for scripts.
- The documentation, libraries, and any other materials produced under this proposal must be consistently maintained and kept up-to-date.

Key Proposal Deliverables

1. The specification of the tools.
2. Web Interface UI
3. A mechanism for saving ScriptContexts and executing them;
4. A mechanism for saved data export
5. A test-suite that covers the aforementioned tools
6. Ecosystem integration: Docker containers, documents, configuration files, etc.

Milestones

Milestone	Deliverables	Acceptance criteria	Weeks
T9.1 Tool Specification and Design	CIP is submitted and reviewed by CIP editors.	Specification is published in the repository.	4

	Proposal reaches accepted status. Specification is complete and usable by external teams.	All planned features and workflows are clearly described. Proposed changes are reflected in planned pull requests.	
T9.2 Web Interface for Script Monitoring and Analysis	Web application enabling: - monitoring of live DApp script execution. - inspection of script inputs, outputs, and execution context. Integration with PSR backend for real-time data access.	The web interface is functional and connected to live or recorded execution data. Users can inspect script execution details through the UI. Public demo showcases end-to-end usage of the interface.	12
T9.3 Runtime Script Substitution Management	Additional endpoints together with frontend UI functionality added to the PSR HTTP API which would allow for CRUD modifications of target and shadow scripts, developed as PRs against the PSR.	CRUD operations for scripts are available via API and UI. Changes take effect without restarting the system. Public demo showcases end-to-end usage of the interface.	4
T9.4 Script Re-Execution with Modified Contexts	Additional endpoints added to the PSR HTTP API which would allow running scripts against modified ScriptContexts.	Scripts can be executed against stored contexts with modified parameters. Results are returned and inspectable via API or UI. Implementation is delivered as reviewed pull requests.	4
T9.5 Export and Reuse of Execution Data	Additional endpoints added to the PSR HTTP API which would allow exporting saved data; Additional executables can be provided for similar purposes.	Execution data can be exported in a usable format. Exported data can be reused outside the tool.	4
T9.6 Test Suite and Validation Infrastructure	A robust test suite. Potentially also: requests to the Consensus Team and/or Node	Tests are implemented and merged into relevant repositories.	16

	Team to tweak their libraries' stable interface.	All core functionality is covered by automated tests. Tests pass consistently in CI.	
T9.7 Developer Documentation and Ecosystem Integration	Public Cardano user resources will include use case scenarios, a cookbook, and integration instructions for the tool with other debug utilities (e.g., cooked-validator), Docker containers are provided, etc.	Documentation is complete and publicly accessible. The tool can be installed and used following documentation only. Documentation is reviewed and accepted by relevant teams.	6