Implementation of a Local Polygon Private Chain

Overview

The implementation of a local private chain based on Polygon using the Erigon and Heimdall clients exposed numerous technical hurdles and substantial gaps in available documentation, necessitating a detailed and comprehensive analysis to identify root causes and potential resolutions. This project involves deploying a blockchain environment that mirrors a production setup but operates entirely within a local infrastructure, allowing for controlled testing and evaluation of its core components. The endeavor revealed critical limitations in existing tools and configurations, underscoring the need for improved modularity, synchronization, and user guidance.

Key Issues

1. Communication Between Layers

The most significant issue identified was the lack of proper communication between the consensus layer (Heimdall) and the execution layer (Erigon/Bor). This breakdown in interaction has both functional and operational implications:

- **Absence of milestone production by Heimdall:**
    - Specifically, Heimdall fails to generate milestones, causing RPC call errors from Bor to Heimdall. For instance, attempts to query the `/milestone/latest` endpoint result in a 500 error with the following message:
      {"error":"{"codespace":"sdk","code":1,"message":"could not fetch milestone; ERROR:\nCodespace: 1\nCode: 7501\nMessage: \"Milestone Not Found\"\n\"}"}
    - Logs in Erigon/Bor confirm this failure with warnings such as:
      WARN [01-27|17:46:25.432] Failed to fetch latest milestone
    - Similar issues are observed with the `/milestone/lastNoAck` endpoint, which returns code 200 but produces the following log warning:
      WARN [01-28|17:38:57.414] Failed to fetch latest no-ack milestone.
      Endpoint output:
      {"height": "521","result": {"result": ""}}

- **Blocks produced only on the consensus layer but not on the execution layer:**
    - Test transactions sent to one of the running nodes are received but not processed or mined. This reflects a fundamental disconnect between Heimdall and Bor.
    - While the execution layer establishes a connection with Heimdall CL, block production remains absent, highlighting synchronization and logic gaps.

2. Genesis Configuration

The process of configuring the genesis block presented substantial difficulties due to the absence of robust tools and standardized procedures:

- **Execution Layer (EL):**
    - A notable lack of utility tools for the automated configuration of genesis files added complexity to the setup.
    - Parameters needed to be manually configured, introducing the risk of human error and inconsistency.
- **Consensus Layer (CL):**
    - The `heimdalld` tool was utilized for configuration; however, synchronizing its parameters with the execution layer proved non-trivial.
    - The lack of clear integration mechanisms complicated efforts to achieve a cohesive configuration across both layers.

3. Documentation Gaps

The absence of cohesive and comprehensive documentation significantly impeded the implementation process:

- No detailed guides were available for running Erigon and Bor in a local environment.
- Best practices for initial setup were missing, leaving users to rely on fragmented information from disparate sources.
- The lack of illustrative examples further complicated the process of parameter configuration for local node execution, consuming significant time and effort.

Implemented Setup

Configuration Details

- **Genesis Block:**
    - A genesis block was created with two validators to simulate a basic network.
    - Smart contracts were partially implemented, starting with the core contracts from Amoy's genesis configuration:
        - `state_receiver`
        - `validator_set`
        - Excluded contracts include `staking`, `slash`, and other auxiliary components.
- **Node Configuration:**
    - Validation accounts were established on two Erigon/Bor nodes.
    - Connections between nodes were successfully set up using enode identifiers.

Current Limitations

1. **Post-genesis Forks:**
    - Potential incompatibilities with Amoy's configuration may arise during post-genesis operations.
    - The management of forks following the genesis block remains an unresolved challenge.
2. **Setup from Scratch:**
    - A full rebuild of the chain may be required, as the current implementation does not execute properly from the genesis block.
    - Replicating the Amoy environment in its entirety poses significant challenges due to its complexity and lack of documentation.

Implemented Solutions

Documentation and Scripts

To address the challenges and improve reproducibility, the following were developed:

1. Automated scripts to streamline the setup process and minimize manual configuration.
2. A detailed README document outlining step-by-step procedures for setting up the environment.
3. Comprehensive documentation of the encountered issues and their respective troubleshooting steps, providing a valuable resource for future implementations.

Recommendations for Improvement

1. **Utility Tools:**
    - Develop robust tools to automate genesis configuration, reducing manual effort and errors.
2. **Enhanced Documentation:**
    - Create focused and detailed guides for setting up and operating Erigon,Bor and Heimdall in local environments.
    - Include best practices and illustrative examples to aid users in navigating complex configurations.
3. **Automation:**
    - Implement end-to-end automated procedures for deploying the entire setup, ensuring consistency and efficiency.

Conclusions

The implementation of a local Polygon private chain revealed significant technical and procedural challenges, particularly in the areas of layer synchronization and initial configuration. While the produced documentation and scripts represent a meaningful step forward, much work remains to achieve a streamlined and reliable process.

The most pressing challenges involve improving communication between the consensus and execution layers, as well as developing strategies to manage post-genesis forks effectively. Addressing these issues will require further research and architectural innovation to create a scalable and user-friendly solution. The insights gained from this project provide a valuable foundation for enhancing the Polygon ecosystem and its adaptability to localized environments.