Coordinated checkpointing of distributed applications

[criu.org]

Mentors:

- Radostin Stoyanov <radostin.stoyanov@eng.ox.ac.uk>
- Prajwal S N <prajwalnadig21@gmail.com>

Mentee personal information:

- Name: Kouamé Behouba Manassé
- Email: behouba@gmail.com
- GitHub: github.com/behouba
- Matrix/Gitter: @behouba:matrix.org
- Location: Montreal, QC, Canada
- Timezone: GMT-4

Problem statement

Kubernetes has recently introduced a checkpointing functionality that allows for transparently capturing and restoring the runtime state of containers. This functionality works in a similar manner to container checkpointing with engines such as Docker and Podman. However, in contrast to these container engines, Kubernetes is often used to automate the deployment, scaling, and management of distributed applications. Checkpointing such distributed workloads (e.g., ML training) requires a synchronization mechanism with multiple CRIU instances that potentially run on different cluster nodes.

The criu-coordinator tool aims to provide such a synchronization mechanism using CRIU's action-script hooks. However, one of the key challenges with this mechanism is the handling of network connections.

In particular, Kubernetes containers run within network namespaces allocated to Pods, and container checkpoint/restore is performed within these namespaces. This approach prevents the use of the traditional network locking mechanism provided by CRIU.

Proposed solution

The goal of this project is to extend criu-coordinator with support for handling of established TCP connections when checkpointing distributed applications. The proposed solution will be implemented in several phases:

- **Phase 1**: Extend criu-coordinator with **network-lock** and **network-unlock** hook handlers with criu-coordinator.
- **Phase 2**: Implement a Pod-level network locking mechanism that enables checkpointing and restoring of distributed applications with established TCP connections.
- **Phase 3**: Extend the coordinated checkpointing mechanism to differentiate between **internal** and **external** TCP connections.
    - **Internal connections** are those between containers that are being checkpointed, and we save/restore both ends of the connection.
    - **External connections** are those where only one end of the connection is checkpointed, and they can be restored in a "**closed**" state.

Implementation plan and timeline

1. **Preparation (Before May 8):**
    - Familiarize myself with criu-coordinator codebase, CRIU action-script hooks, and Kubernetes checkpointing capabilities.
    - Set up a testing environment to manually checkpoint distributed applications.

2. **Design and planning (May 8 – June 2):**
    - Discuss design and new workflow with mentors.
    - Define testing and validation scenarios.
    - Produce a design document of design choice and workflow for the integration of network-lock/unlock hooks, namespace tracking, and restore mechanisms.

3. **Implementation of network locking (June 2 - June 27):**

    Weeks 1-4:

    - Develop and integrate the support of **network-lock** and **network-unlock** hooks in criu-coordinator client-server logic.
    - Implement pod-level network state tracking for distributed containers.
    - Prototype a mechanism to capture the network state during checkpointing.
    - Extend the existing server logic to coordinate network locks across dependent distributed containers.

4. **Restoration workflow and state reconciliation (June 30 - July 25):**

   Weeks 5-8:

   - Develop and integrate support for **post-restore** hooks to manage network reattachment (**network-unlock** and **pre-resume** hooks).
   - Implement revalidation mechanisms to ensure that restored connections match the pre-checkpoint state.
   - Test and iterate on connection reattachment workflows.

5. **Improving network state management (July 28 - August 15):**

   Week 9-11:

   - Implement a logic to distinguish between internal and external TCP connections.
   - Define strategies for resolving network state conflicts and cleaning up stale connections.

6. **Testing, Optimization, and Documentation (August 18 - September 1):**

   Week 12-13:

   - Perform tests on real-world distributed applications to benchmark performance and reliability.
   - Validate the solution using distributed applications such as databases and message queues.
   - Prepare documentation, a demo, and finalize optimizations based on feedback.

7. **Post-GSoC2025 (after September 1):**
   - Address any remaining issues, incorporate mentors and community feedback,
   - Support ongoing maintenance and future enhancements.

About me

I am currently a Master's student in Computer Science at Polytechnique Montréal. I participated in GSoC 2023 with CRIU, where I collaborated with Adrian Reber, Radostin Stoyanov, and Prajwal S N to implement forensic analysis features in checkpointctl. Details of my previous work on go-criu and checkpointctl, are available in my GSoC 2023 final report. I am also currently a maintainer of checkpointctl and have experience working with containers and Kubernetes.

I am motivated to contribute to this project, and I believe my skills and experience will help implement coordinated checkpointing for distributed applications. Finally, I will not have any courses during the summer and am available to work full-time on this project.