УДК 004.9

ИССЛЕДОВАНИЕ СОВРЕМЕННЫХ МЕТОДОВ ВЫПОЛНЕНИЯ ЗАДАЧ В РАСПРЕДЕЛЕННОЙ СРЕДЕ ОБЛАЧНЫХ ВЫЧИСЛЕНИЙ И РАЗРАБОТКА АРХИТЕКТУРЫ СИСТЕМЫ ДЛЯ РАСПРЕДЕЛЕННОГО РАСЧЕТА СИСТЕМНО-ДИНАМИЧЕСКИХ МОДЕЛЕЙ В ОБЛАКЕ

Шагиев С. И. (Университет ИТМО)
Научный руководитель – кандидат технических наук, Перл И.В. (Университет ИТМО)

Введение. В условиях быстрого роста облачных технологий и широкого использования контейнеризации в современных вычислительных системах, вопросы эффективного масштабирования и надежности вычислительных контейнеров становятся критически важными. В частности, для увеличения отказоустойчивости приложений, осуществляющих длительные вычислительные операции в контейнерах, необходимо решить проблемы, связанные с автоматическим выделением ресурсов, балансировкой нагрузки и устойчивостью к возникающим ошибкам. Одним из таких приложений является платформа sdCloud, архитектура которой позволяет исполнять очень ограниченное число системно-динамических моделей параллельно, а также не приспособлена к восстановлению аварийно завершившихся симуляций.

Основная часть. Для решения вышеуказанных проблем были проведены сравнительные исследования современных подходов и инструментов.

Множество современных инструментов позволяет организовать автоматическое масштабирование контейнеризированных приложений. Популярная система Kubernetes предоставляет широкий функционал по управлению контейнерами, однако имеет недостаток в виде сложности настройки и поддержки [1]. Зачастую необходим отдельный специалист, ответственный за работу с Kubernetes, что неудобно для организаций, не имеющих возможности содержать такого специалиста, например, для небольших компаний или лабораторий. Потребности таких организаций может удовлетворить Docker Swarm, являющийся стандартным решением для управления Docker контейнерами. Однако Docker Swarm предоставляет возможность только ручного управления количеством подобных контейнеров. Для настройки автоматического масштабирования необходимо использовать инструменты, дополняющие Docker Swarm, такие как Elixir [2].

Эффективное решение проблемы устойчивости к возникающим ошибкам можно найти в архитектуре фреймворка Apache Spark, позволяющего сохранять прогресс вычисления в контрольные точки и восстанавливать вычисления из них. Однако использование данного фреймворка приносит необходимость писать программу специально под него с нуля, либо вносить существенные изменения в исходный код существующего приложения. Также использование Apache Spark ограниченно набором поддерживаемых языков программирования. Вышеописанных проблем лишен подход с использованием инструмента CRIU, позволяющего создавать слепки linux процессов, исполнение которых можно продолжить в будущем. Такой слепок можно создать для любого процесса, вне зависимости от конкретной реализации и используемого языка программирования [3-4]. Позже этими слепками можно управлять как обычными файлами, в том числе перемещать их по сети между узлами облачного кластера [5].

На основе исследований была разработана архитектура решения, позволяющего осуществлять автоматические мониторинг и масштабирование контейнеров в кластере под управлением Docker Swarm с применением подходов, используемых в Elixir. Elixir устанавливает дополнительный Swarm, цель которого заключается в мониторинге существующих Swarm'ов и балансировке нагрузки между узлами, запускающими одно и то же приложение (то есть узлами, принадлежащими к одному и тому же Swarm). Пороговые значения, при которых осуществляется увеличение или уменьшение количества запущенных

контейнеров, задаются в конфигурации. Также внутри каждого контейнера исполняется экземпляр сервиса, ответственного за периодическое сохранение состояний процессов, выполняющих задачи с длительным временем исполнения, в слепок с помощью CRIU, за отправку этого слепка в отказоустойчивое хранилище внутри кластера, а также за восстановление аварийно завершившихся процессов из сохраненных слепков. Такие параметры как периодичность сохранения и местоположение хранилища задаются в конфигурационных файлах.

На основе предложенной архитектуры было разработано приложение, осуществляющее управление контейнерами с исполнителями системно-динамических моделей для платформы sdCloud. Приложение решило проблему, которая заключалась в том, что все вычисления производились в одном контейнере, и когда приходило много заявок на исполнение, у контейнера с исполнителем заканчивались свободные ресурсы, он не справлялся с нагрузкой и аварийно завершал работу. Эксперименты показали, что возможное количество одновременно исполняемых моделей возросло в 5 раз. А также была решена проблема, при которой после ошибки во время выполнения модели терялся весь прогресс, и исполнение модели приходилось запускать с самого начала. Эксперименты показали уменьшение среднего времени исполнения моделей на 10%.

Выводы. Были проанализированы существующие подходы для автоматизации масштабирования контейнеризированных приложений и сохранения и восстановления прерванных вычислений. Была создана архитектура решения, на основе которой было разработано приложение для платформы sdCloud, увеличившее возможное количество одновременно исполняемых моделей, увеличившее надежность вычислений и уменьшившее среднее время исполнения моделей.

Список использованных источников:

1. Vayghan, Leila Abdollahi, Mohamed Aymen Saied, Maria Toeroe and Ferhat Khendek. Kubernetes as an Availability Manager for Microservice Applications. ArXiv abs/1901.04946 (2019)
2. Alexiou, M.S., Petrakis, E.G.M. (2020). Elixir: An Agent for Supporting Elasticity in Docker Swarm. In: Barolli, L., Amato, F., Moscato, F., Enokido, T., Takizawa, M. (eds) Advanced Information Networking and Applications. AINA 2020. Advances in Intelligent Systems and Computing, vol 1151. Springer, Cham. https://doi.org/10.1007/978-3-030-44041-1_96
3. Vasyukov, A., & Beklemysheva, K. (2018). Using CRIU with HPC Containers Field Experience. International Journal of Engineering and Computer Science, 7(07), 24106–24108. https://www.doi.org/10.18535/10.18535/IJECS/V7I7.01
4. Hein Htet, Nobuo Funabiki, Ariel Kamoyedji, Xudong Zhou, and Minoru Kuribayashi. 2021. An Implementation of Job Migration Function Using CRIU and Podman in Docker-based User-PC Computing System. In Proceedings of the 9th International Conference on Computer and Communications Management (ICCCM '21). Association for Computing Machinery, New York, NY, USA, 92–97. https://doi.org/10.1145/3479162.3479176
5. Sindi, Mohamad, and John R. Williams. "Using container migration for HPC workloads resilience." 2019 IEEE High Performance Extreme Computing Conference (HPEC). IEEE, 2019.