Подход к развертыванию программного обеспечения вычислительного кластера по облачной модели

Роман Костромин

Институт динамики систем и теории управления им. В.М. Матросова СО РАН, ул. Лермонтова, д. 134, 664033, Иркутск, Россия

Аннотация

В работе обсуждается модернизация модели управления вычислительным кластером с использованием контейнеризации для расширения возможностей суперкомпьютерного центра. Предлагаемый подход позволяет сохранить традиционные способы управления ресурсами кластера и внедрить новые сценарии использования. Ожидается, что сочетание методов и средств контейнеризации позволит существенно ускорить подготовку и проведение научных исследований, снизить затраты на содержание инфраструктуры и улучшить общую полезную загрузку кластера. Предлагаемая модель разрабатывается с расчетом на привлечение новых исследователей и разработчиков за счет расширения возможностей использования вычислительных ресурсов и внедрения передовых технологий.

Ключевые слова

Вычислительный кластер, администрирование, автоматизация развертывания операционных систем, контейнеризация

1. Введение

В последние годы наблюдается значительная тенденция к расширению сферы применения суперкомпьютеров, включая решение новых задач из области машинного обучения, искусственного интеллекта, микросервисного подхода и других передовых технологий [1, 2]. Это расширение подчеркивает не только сохраняющуюся актуальность суперкомпьютеров, но и их растущее значение в качестве мощного инструмента для развития современных научных и технологических направлений. Благодаря этому академические суперкомпьютерные центры (СКЦ) становятся ключевым элементом в реализации новых проектов и исследований, подтверждая свою роль как важного актива в достижении прогресса в многочисленных областях науки и техники [3].

Традиционно, СКЦ представляют собой комплексные системы, поставляемые специализированными подрядчиками и требующие значительных усилий для поддержания их работоспособности. В состав таких систем входят разнообразные инструменты и программное обеспечение (ПО), включая средства для параллельного выполнения команд, мониторинга, оповещения, а также для развертывания базовых образов операционных систем (ОС). Однако, существующие средства автоматизации часто ограничиваются поддержкой исходного состояния узлов, рекомендованного поставщиками, что влечет за собой ограничения на спектр выполняемых научных приложений и не предоставляет инструментов для пользователей, чьи задачи требуют специализированного окружения и определенных версий ПО.

С точки зрения решения упомянутых выше проблем, перспективным направлением является применение методов и средств управления ресурсами основанных на виртуализации и контейнеризации, применяемых в центрах обработки данных и у облачных провайдеров [1, 2].

6th International Workshop on Information, Computation, and Control Systems for Distributed Environments (ICCS-DE 2024), July 01–05, 2024, Irkutsk, Russia
EMAIL: roman@kostromin.net (A. 1)
ORCID: 0000-0001-8406-8106 (A. 1)

ICCS-DE 2024 Workshop Proceedings
DOI: 10.47350/ICCS-DE.2024.18

К сожалению, особенности СКЦ, ориентированных на выполнение параллельных MPI приложений, накладывают ограничения на применение подходов, традиционных для облачных ресурсов и обоснованно требуют проведения дополнительных исследований в этой области.

Таким образом, в данной работе обсуждаются вопросы модернизации классического подхода управления вычислительным кластером (ВК) с целью интеграции методов и средств из сферы управления облачными ресурсами. Предложены основные принципы обновленного подхода, а также ключевые его компоненты. Прототип данного подхода был протестирован на резервных узлах Центра коллективного пользования «Иркутский суперкомпьютерный центр СО РАН» [4] и продемонстрировал реализуемость данного проекта и соответствие предъявляемым администраторами требованиям.

2. Особенности управления ВК

Классический цикл работы с ВК со стороны пользователя включает следующие этапы [5]:

- Удаленный вход на кластер;
- Копирование данных между ВК и компьютером пользователя;
- Редактирование исходных текстов программ;
- Компиляция программ;
- Запуск задач и работа с очередью;
- Копирование результатов вычислений;
- Завершение сеанса.

Рисунок 1: Типовая архитектура ВК с точки зрения пользователя

Иными словами, пользователь формирует паспорт задания с учетом регламента и квот доступа к кластеру, ставит его в очередь локального менеджера ресурсов (ЛРМ) через узел доступа к кластеру (см. Рисунок 1). В соответствии с политикой диспетчирования задания распределяются по вычислительным узлам. Узлы взаимодействуют между собой в процессе вычислений по высокоскоростной сети, такой как Infiniband или аналогичной. С этой точки зрения ВК воспринимается как черный ящик, способы взаимодействия с которым строго регламентированы. Например, установка дополнительного ПО допустима только в домашний каталог пользователя силами самих пользователей.

Иной взгляд на ВК имеют его администраторы (см. Рисунок 2). Такая архитектура ВК встречается повсеместно в СКЦ, поставляемых российскими суперкомпьютерными компаниями. В ней выделяется ключевая роль виртуальной машины (ВМ) master, для которой настроено реплицирование файловой системы виртуального диска через технологию DRBD на два сервера (master-1 и master-2). ВМ master управляет сетью ВК, вычислительными узлами, содержит локальный менеджер ресурсов (ЛМР), каталоги пользователей LDAP, систему мониторинга и т.д. Доступ к кластеру осуществляется через основной и резервный узлы доступа (access-1 и access-2). Все системы подключены по высокоскоростной сети к производительной системе хранения данных (СХД).

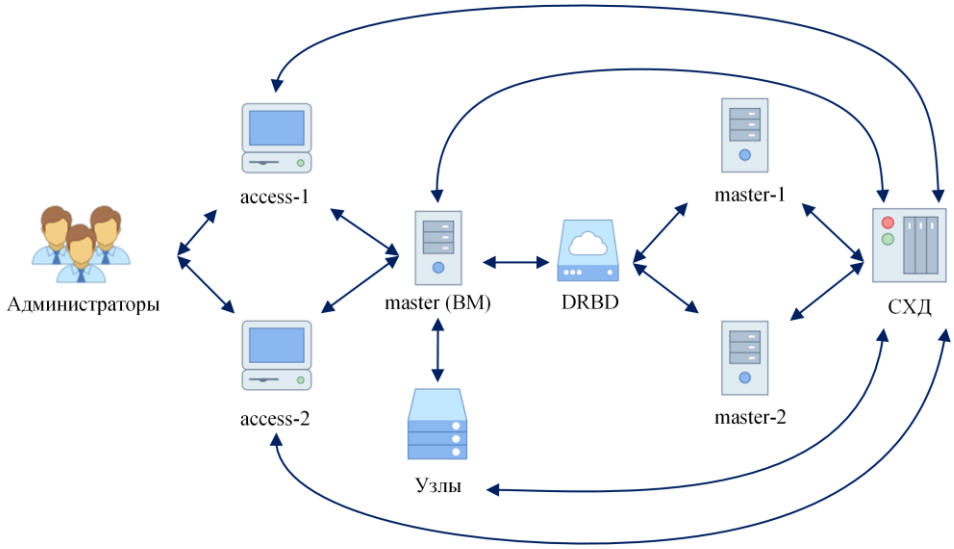

Рисунок 2: Типовая архитектура ВК с точки зрения администратора

Рассматриваемая архитектура хорошо зарекомендовала себя в течение длительного времени эксплуатации ВК, но при этом выявлены следующие недостатки, а именно:

- повреждение структуры DRBD или виртуального диска master приводит к выходу из строя ВМ master;
- при выходе из строя ВМ master затрудняется управление кластером и его использование из-за проблем синхронизации SSH и LDAP;
- при выходе из строя СХД данных затрудняется управление кластером и его использование (нет каталогов пользователей, библиотек и программ);
- узлы master-1, master-2, access-1 и access-2 большую часть времени простаивают.

Как показала практика, конфигурация с DRBD может длительное время работать надежно, но в случае сбоев в работе ВМ master восстановление и устранение неполадок занимает значительное время, что ведет к простою работы ВК. Кроме того, регулярные запросы пользователей о возможности установки дополнительных системных и прикладных библиотек, отсутствующих на кластере, актуализируют задачу модернизации существующей модели работы ВК. Таким образом, несмотря на достоинства архитектуры классического ВК, необходимо принять меры по устранению выявленных недостатков, чтобы обеспечить более стабильное и эффективное функционирование кластера в будущем, а также создать условия для привлечения новых пользователей.

Для решения этих проблем облачные провайдеры предлагают два подхода – применение виртуализации и контейнеризации. Преимущества виртуализации управляющей системы заключаются в том, что благодаря разделению ролей ВМ master на несколько независимых ВМ, снижаются риски одновременного отказа всех компонентов управляющей системы. При этом отказоустойчивость обеспечивается созданием кластера виртуализации с применением нескольких физических серверов с конфигурированием живой миграции и реплицированния ВМ между узлами. Применение виртуализации целесообразно для разделения управляющих компонентов на несколько независимых виртуальных сущностей, однако для вычислительных узлов виртуализация влечет значительные накладные расходы [6].

Как показано в работах [7-8], популярным и устоявшимся подходом к построению изолированных вычислительных сред является контейнеризация ПО. В частности, такая контейнеризация осуществляется с помощью Docker. В отличие от виртуализации, контейнеризация снижает затраты на подготовку, хранение и развертывание настроенных образов ОС и ПО [8]. Однако Docker это не лучшее решение для высокопроизводительных сред, в отличие от Singularity, которая обеспечивает полную поддержку образов Docker и устраняет недостатки Docker для запуска контейнеров с точки зрения производительности и обеспечения безопасности вычислений [9].

Для управления группой узлов в среде контейнеризации (оркестрации) традиционно применяют Kubernetes, на его основе и предложена новая модель управления СКЦ (см. Рисунок 3). Ее основой является управляющий кластер на базе Kubernetes, работающий на нескольких узлах управления (УУ). Средствами Kubernetes выполняется запуск образов контейнеров (Docker и Singularity), хранимых в локальной СХД. Часть контейнеров содержат компоненты управления (КУ) кластером, которые выполняются на УУ. Kubernetes создает изолированные сети (виртуальная сеть) для организации взаимодействия между контейнерами вычислительных задач (КВЗ). На вычислительных узлах осуществляется запуск контейнеров Singularty с поддержкой локальных менеджеров ресурсов (PBS, Slurm) в рамках установленных квот пользователей для работы с очередью заданий. Образы, хранимые в СХД подготавливаются администраторами СКЦ заранее и обеспечивают послойную интеграцию необходимых компонентов для приложений и их модулей, запускаемых на ВК.

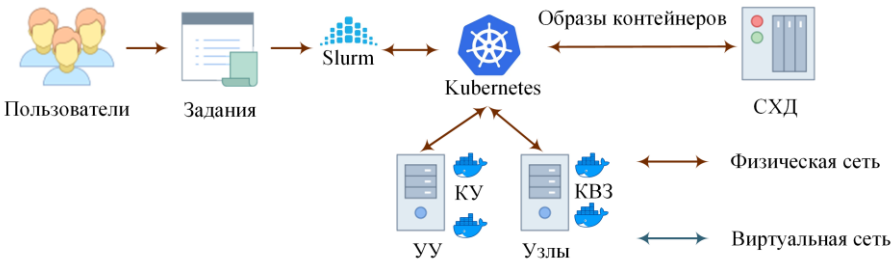

Рисунок 3: Контейнеризация компонентов ВК

3. Вопросы совместимости предложенного подхода с классическим подходом управления ВК

При обновлении модели управления кластером необходимо помнить, что СКЦ ориентирован в первую очередь на пользователей и применение кластера для научных задач не должно быть связано со значительными техническими сложностями. Поэтому необходимо обеспечить как преемственность (поддержку выполнения приложений, несовместимых с новыми системными библиотеками), так и поддержку новых библиотек и технологий. В рамках одной ОС реализовать оба фактора затруднительно. Одним из решений является развертывание (provisioning или deploy) нужной версии ОС и всех необходимых компонентов на узлах, подходящих для задач пользователей. Этот процесс хорошо автоматизирован с помощью специализированных средств (xCAT, OpenHPC, MAAS). Но с точки зрения полезной загрузки ресурсов данный подход несостоятелен, т.к. каждое развертывание занимает порядка 20 минут, что совершенно не применимо для ВК.

По этой причине стоит рассматривать узлы в режиме stateless, т.е. ОС узла не имеет определенного состояния и настроек, загружается по сети и не требует для своей работы физического накопителя, является легковесной и занимает минимальный объем оперативной памяти (до 500 МБ). Развертывание такой системы занимает порядка 5 минут, из которых до 80% затрачивается на этап загрузки BIOS и самотестирование узла до начала непосредственной загрузки ОС. Основное назначение такой системы – получить от управляющего узла образ контейнера из СХД по высокоскоростной сети, запустить его (10-15 сек.) и передать все доступные ресурсы узла данному контейнеру. Именно Kubernetes реализует управление контейнерами таких бездисковых ОС. Контейнер после завершения работы уничтожается и узел полностью освобождается. В рамках предложенного подхода Kubernetes встраивается в существующий менеджер ресурсов (PBS, Slurm), благодаря чему сохраняется подход с общей очередью заданий и учет квот пользователей для тех, кому в явном виде применение контейнеризации не требуется.

Применение контейнеров позволяет создавать окружение, которое требуется для заданий пользователей. Администраторы СКЦ подготавливают базовые образы систем, содержащие все

необходимые компоненты для разных типов задач (C++ и MPI, Python и CUDA, Hadoop, Apache Ignite и т.д.). Образцы контейнеров могут содержать как устаревшие версии компиляторов и системных библиотек, так и самые современные. Благодаря этому расширяется спектр поддерживаемых языков и системных библиотек, сохраняется привычный способ работы с кластером, снижаются накладные расходы на развертывание ОС на узлах и сокращается количество потенциальных точек отказа системной составляющей кластера.

Для тестирования приложенного подхода было задействовано 10 узлов ВК со следующими характеристиками (см. Таблица 1).

Таблица 1. Характеристики узлов

CPU	RAM	Storage	Network	OS
Intel Xeon E5-2695 v4 «Broadwell» 2.1 GHz, (2 CPU, 72 cores)	128 Gb	256 Gb SSD; 40 Tb, RAID 5	Ethernet, 1 Gbit/s; InfiniBand, 40 Gbit/s	Linux-based

Средствами xCAT на данных узлах была развернута RedHat-совместимая ОС RockyLinux 9.3. Она является продолжателем ОС Centos, чья поддержка завершена в 2024 г. Для запуска MPI-приложений подготовлен базовый образ (БО) контейнера, в который интегрированы требуемые системные модули (СМ) и компоненты ЛМР. На основе БО созданы дополнительные образы для разных классов ресурсов и типов задач. В них дополнительно встроены средства работы с временной базой данных (ВБД), средства интеграции с системой мониторинга и локальным Git-репозиторием.

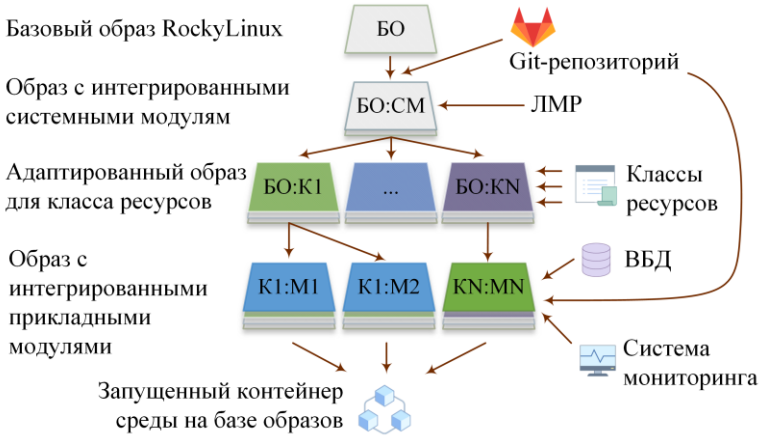

Рисунок 4: Структура слоев контейнеров для ВК

Дополнительно настроена ВМ с Kubernetes и выполнена настройка тестовых узлов для взаимодействия с Kubernetes. Через ЛМР Slurm поставлено в очередь задание, на основании которого средствами Kubernetes выполнено развертывание образов с последующим запуском MPI-задач для оценки производительности узлов (Linpack). Аналогичный набор задач был запущен на данных узлах в явном виде без применения средств контейнеризации. В обоих случаях производительность составила 7.9 Tflops. Тесты производительности Singularity на большем количестве узлов подтверждают, что применение Singularity для ВК не оказывает негативного влияния на общую производительность Tflops [10, 11].

4. Заключение

Применение контейнеризации является востребованным направлением для интеграции в СКЦ. Однако необходимо этот подход к контейнеризации развивать комплексно, обеспечив поддержку личного кабинета пользователя кластера и возможность самостоятельной тонкой настройки базовых образов и т.д., т.к. существующие решения не покрывают всего спектра

применения контейнеризации для ВК. При этом сам подход является гибким и тестирование на небольшом количестве узлов подтверждает его функциональность. Данное исследование находится на начальном этапе, но первые результаты показывают перспективность и достижимость поставленных целей, формируя достаточную основу для дальнейших исследований.

5. Благодарности

Исследование выполнено при поддержке Министерства науки и высшего образования Российской Федерации, проект № FWEW-2021-0005 «Технологии разработки и анализа предметно-ориентированных интеллектуальных систем группового управления в недетерминированных распределенных средах» (рег. № 121032400051-9).

6. Список литературы

[1] А.О. Кудрявцев, В.К. Кошелев, А.О. Избышев, А.И. Аветисян. Высокопроизводительные вычисления как облачный сервис: ключевые проблемы, Параллельные вычислительные технологии (ПаВТ 2013): Труды междунар. науч. конф., Челябинск, Издательский центр ЮУрГУ, 2013, с. 432–438.

[2] S. Deng. Cloud-Native Computing: A Survey From the Perspective of Services, Proceedings of the IEEE, 2023. vol. 112, pp. 12–46. DOI: 10.1109/JPROC.2024.3353855.

[3] С.В. Поляков, А.В. Выродов, Д.В. Пузырьков, М.В. Якобовский. Облачный сервис для решения многомасштабных задач нанотехнологии на суперкомпьютерных системах, Труды ИСП РАН, 2015, Т. 27, № 6, с. 409–420. DOI 10.18522/2311-3103-2016-12-103114.

[4] Иркутский суперкомпьютерный центр СО РАН [Электронный ресурс]. Режим доступа: https://hpc.icc.ru. (Дата обращения 10.06.2024 г.).

[5] С.А. Жуматий, К.С. Стефанов. Суперкомпьютеры: Администрирование, М: МАКС Пресс, 2018.

[6] B.M. Shabanov, O.I. Samovarov. Building the Software-Defined Data Center, Programming and Computer Software, 2019, vol. 45, pp. 458–466. DOI: 10.1134/S0361768819080048.

[7] А.О. Кудрявцев и др. Разработка и реализация облачной системы для решения высокопроизводительных задач, Труды ИСП РАН, 2013, Т. 24, с. 13–34.

[8] M.A. Rodriguez, B. Rajkumar. Container-based cluster orchestration systems: A taxonomy and future directions, Software: Practice and Experience, 2018, vol. 49, pp. 698–719. DOI: 10.1002/spe.2660.

[9] O. Freyermuth, P. Wienemann, P. Bechtle, K. Desch. Operating an HPC/HTC cluster with fully containerized jobs using HTCondor, Singularity, CephFS and CVMFS, Computing and Software for Big Science, 2021, vol. 5, is. 1, pp. 9. DOI: 10.1007/s41781-020-00050-y.

[10] G. Hu, Y. Zhang, W. Chen. Exploring the performance of singularity for high performance computing scenarios, 2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems (HPCC/SmartCity/DSS), IEEE, 2019, pp. 2587–2593.

[11] А.Г. Феоктистов, Р.О. Костромин, М.Л. Воскобойников, Д.И. Ли-Дэ. Организация вычислительной среды разработки и применения научных рабочих процессов на основе контейнеризации, Вычислительные технологии,2023, Т. 28, № 6, с. 151–164.