DOI 10.37882/2223-2966.2023.6-2.22

МОНИТОРИНГ ПОГРАНИЧНЫХ УСТРОЙСТВ НА ОСНОВЕ ТЕХНОЛОГИЙ KUBERNETES

MONITORING OF EDGE DEVICES BASED ON KUBERNETES TECHNOLOGIES

V. Kosenkov

Summary: The concept of «Internet of Things» is widely used in various life applications, from the home to large cities, where there are many peripheral devices to facilitate edge computing. In this article, as a goal, an attempt is made to use Kubernetes to create an environment for collecting the usage status of internal resources of various node devices and implementing monitoring with visualization. Deep learning models are deployed on heterogeneous devices to evaluate and validate performance. In this experiment, Kubernetes used Docker containers to deploy all applications. Then, all the data can be visualized via Grafana or Prometheus.

Keywords: monitoring, Prometheus, Grafana, Kubernetes, Devops.

Косенков Виталий Владимирович
СПБГУТ. проф. М.А. Бонч-Бруевича
vitaliykosein@gmail.com

Аннотация: Концепция «Интернет вещей» широко используется в различных жизненных приложениях, от дома до больших городов, где есть множество периферийных устройств для облегчения граничных вычислений. В этой статье в качестве цели предпринимается попытка использовать Kubernetes для создания среды для сбора состояния использования внутренних ресурсов различных узловых устройств и реализации мониторинга с визуализацией. Модели глубокого обучения развертываются на разнородных устройствах для оценки и проверки производительности. В этом эксперименте Kubernetes использовал контейнеры Docker для развертывания всех приложений. Затем, все данные могут быть визуализированы через Grafana или Prometheus.

Ключевые слова: мониторинг, Prometheus, Grafana, Kubernetes, Devops.

Сбор данных — это систематический процесс сбора, мониторинга и анализа конкретной информации для поиска решений соответствующих запросов и оценки результатов. Независимо от того, будут ли собранные данные использоваться для проведения исследований в деловой или академической областях, они позволяют получить четкое представление и узнать из первых рук о проблеме исследования

Развитие концепции «Интернет вещей» в последние годы привело к широкому использованию различных передовых вычислительных устройств в различных областях для содействия сбору и предварительной обработке данных. Однако из-за различий в архитектуре, вызванных разнородными периферийными вычислительными устройствами, эти устройства испытывают особые трудности при оценке производительности и мониторинге. С увеличением количества различных периферийных вычислительных устройств мы пытаемся разработать среду мониторинга производительности, которая мгновенно контролируется и проста в обслуживании. В этой среде будут использоваться контейнеризированные методы управления инструментами путем сбора и визуализации различных ресурсов и системных индикаторов в устройствах [1].

Операционная система и виртуализация приложений, также известная как виртуализация на основе контейнеров, стала широко использоваться с 2013 года с запуском проекта с открытым исходным кодом Docker и растущим интересом облачных и интернет-провайдеров. Контейнер — это программная среда, в которой можно развернуть приложение и его компоненты вместе со всеми необходимыми зависимостями, двоичными файлами и базовая конфигурация, необходимая для запуска приложения. Контейнеры обеспечивают более высокий уровень абстракции для управления жизненным циклом процесса с возможностью не только запуска/остановки, но также беспрепятственного обновления и выпуска новой версии контейнерной службы. Кроме того, архитектурные преимущества, предоставляемые контейнерами, позволяют разработчикам использовать приложения или службы в автономных виртуальных средах — задача, которая ранее была прерогативой виртуальных машин.

Что касается среды управления и развертывания, контейнеры Docker просты в развертывании и обладают функциями, подходящими для вычислительных устройств с более низкой эффективностью. Kubernetes используется для управления и развертывания эффективности и жизненного цикла Контейнеры Docker, поддерживающие всю среду в высокодоступном состоянии [2]. Что касается сбора данных об устройствах, в данной статье оценивается развертывание Prometheus как инструмента для сбора и хранения данных разнородных устройств [3]. Prometheus собирает данные с помощью node exporter в качестве агента и устанавливается он на каждое пограничное устройство.

Kubernetes

У Kubernetes есть ценная функция, которая эффективно используется для управления микросервисами и наблюдением за неисправностями [4, 8]. Масштабируемые контейнеры Linux автоматически поддерживают

контейнер всегда с наилучшей производительностью. По сравнению с ручным развертыванием контейнеров на нескольких машинах, Kubernetes может использовать Master в качестве контейнеризованного сервиса распределения и управления подчиненными устройствами во всем кластере, устанавливая кластерную связь с несколькими узлами [5].

Docker

Docker — это программное обеспечение с открытым исходным кодом, открытая платформа для разработки, развертывания и выполнение приложений. Docker позволяет пользователям разделять приложения в системной среде для формирования контейнеров меньшего размера, тем самым увеличивая скорость развертывания программного обеспечения. Контейнеры Docker похожи на виртуальные машины, но в принципе контейнеры виртуализируют операционные системы, а виртуальные машины виртуализируют аппаратное обеспечение, поэтому контейнеры более портативны, чем виртуальные машины, и потребляют меньше системных ресурсов [6].

Prometheus

Prometheus — это программное обеспечение с открытым исходным кодом для мониторинга окружающей среды и оповещения. Оно записывает показатели в режиме реального времени в базу данных временных рядов (TSDB), созданную с использованием HTTP извлекаемая модель и обладает гибкими функциями запроса и мгновенного оповещения.

Node Exporter

Node Exporter — это источник данных Prometheus. Он захватывает различные индексные данные в режиме реального времени и создает модель HTTP pull, чтобы предоставить Prometheus для сбора и хранения.

DeepStream

DeepStream — это инструмент на базе NVIDIA, который в основном используется для решения всего визуального процесса. Она отличается от других визуальных библиотек (таких как OpenCV) тем, что предоставляет полное комплексное решение для поддержки.

Визуализация в Grafana

Grafana — это программное обеспечение с открытым исходным кодом для мультиплатформенного анализа и интерактивной визуализации. Когда он подключается к поддерживаемому источнику данных, он будет предоставлять диаграммы, графические изображения и оповещения для Интернета. Он может быть расширен с помощью подключаемой системы. Пользователи могут использовать интерактивный конструктор запросов для создания сложных информационных панелей мониторинга.

Проектирование и развертывание системы

В этой статье в основном используется сервер в качестве хоста для служб управления и распространения. Его технические характеристики — процессор Intel (R) Core (TM) i7-5960X с тактовой частотой 4,00 ГГц, 128 ГБ оперативной памяти и жесткий диск HDD емкостью 2 т. В качестве рабочих узлов для мониторинга производительности используются четыре периферийных вычислительных устройства, а именно Raspberry Pi 3, Raspberry Pi 4 (4 ГБ), NVIDIA Jetson Nano и NVIDIA Jetson TX2. Экспортер узлов будет собирать внутренние системные индикаторы каждого устройства, а затем отслеживать экспортер узлов каждого устройства через Prometheus, чтобы возвращать данные индикатора и сохранять их в TSDB.

Затем получит доступ база данных Prometheus через Grafana для получения данных, на рис. 1 описана архитектура системы. Prometheus, node exporter и Grafana — все они развертываются службой развертывания контейнеров Docker. Kubernetes использовался для создания кластерной системы. Узел сервера используется в качестве главного (Master) узла. Остальные периферийные вычислительные устройства используются в качестве рабочих узлов для распределения служб и планирования — наименьшего устройства для развертывания служб в модуле. На рисунке 2 показана архитектура кластера Kubernetes.

Сбор показателей ресурсов

Каждое узловое устройство использует экспортер узлов для сбора показателей внутренних ресурсов. Разные устройства должны использовать отдельный экспортер узлов из-за разных архитектур [7]. После завершения установки каждый индекс ресурса, полученный в соответствии с внутренними настройками, будет экспортирован через порт (9100). Производными показателями ресурсов являются загрузка процессора, памяти, загрузка системы и т.д. После сбора и хранения данных, а также после создания всей среды мониторинга, необходимо проверить, может ли цель мониторинга использования ресурсов быть достигнута, в этом эксперименте используются модели глубокого обучения на двух гетерогенных узлах пограничных устройств, Raspberry Pi 4 и NVIDIA Jetson Nano.

Сбор и визуализация данных

Что касается сбора индексов данных, Prometheus добавил четыре разнородных устройства для монито-

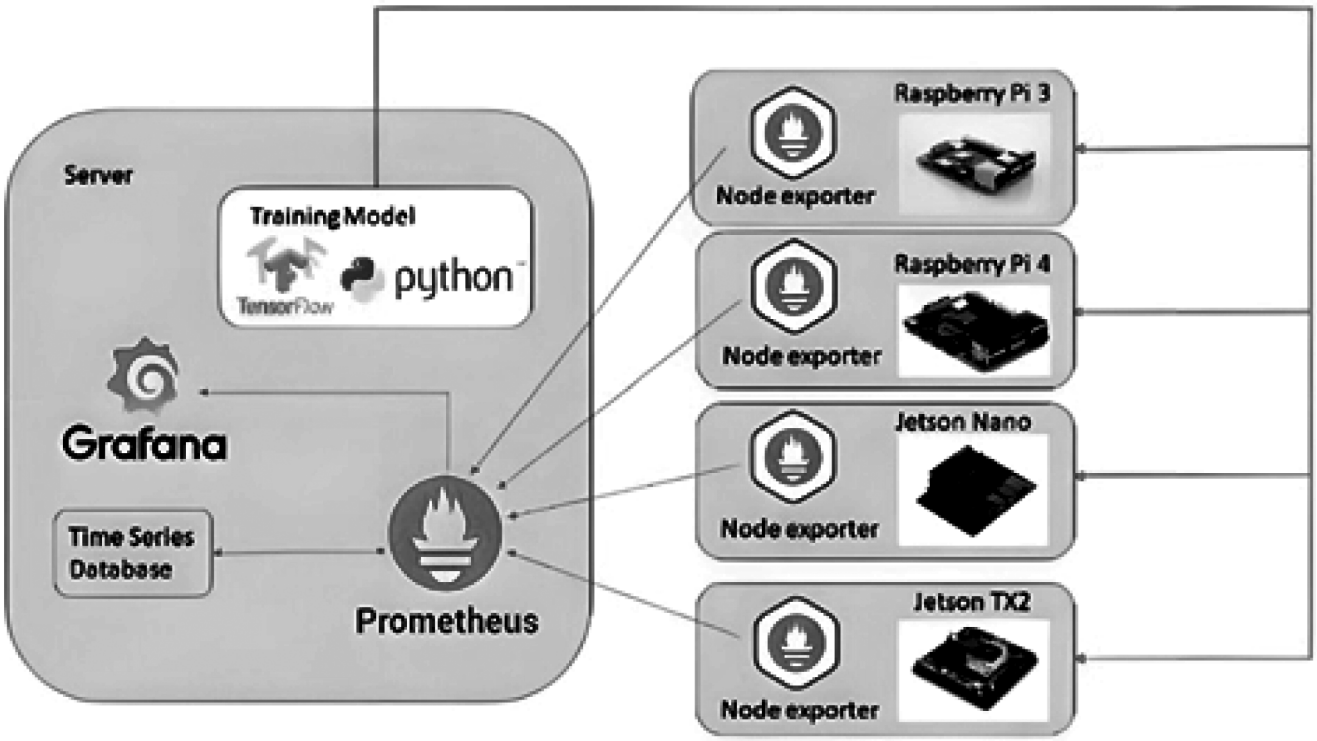


Рис. 1. Схема архитектуры системы

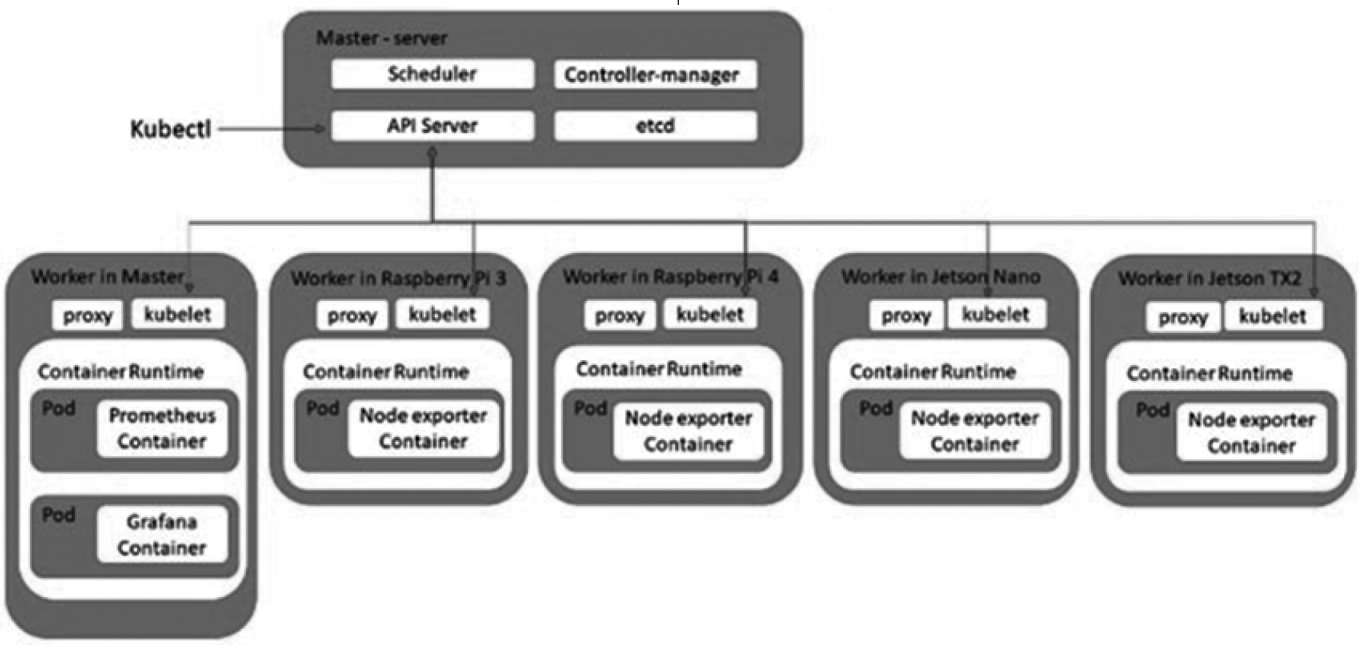


Рис. 2. Кластерная архитектура Kubernetes

ринга, включая NVIDIA Jetson Nano, NVIDIA Jetson TX2, Raspberry Pi 3 и Raspberry Pi 4 в качестве устройств мониторинга ресурсов, как показано на рис. 3. Среди них Raspberry Pi 4 и NVIDIA Jetson Nano являются объектами мониторинга этого изменения ресурсов. Экспериментальная проверка ресурсов с точки зрения изменений в Raspberry Pi 4, выполненная с помощью модели Tensorflow trained Inception v3, которая представляет собой идентификацию из модели транспортного средства, соответственно, выполнен FP16. На рис. 4a и рис. 4b показано выполнение FP16 продольного изменения ресурсов процессора и памяти.

Заключение

В этой статье Kubernetes использовался для создания высококачественной среды мониторинга производительности пограничного гетерогенного оборудования

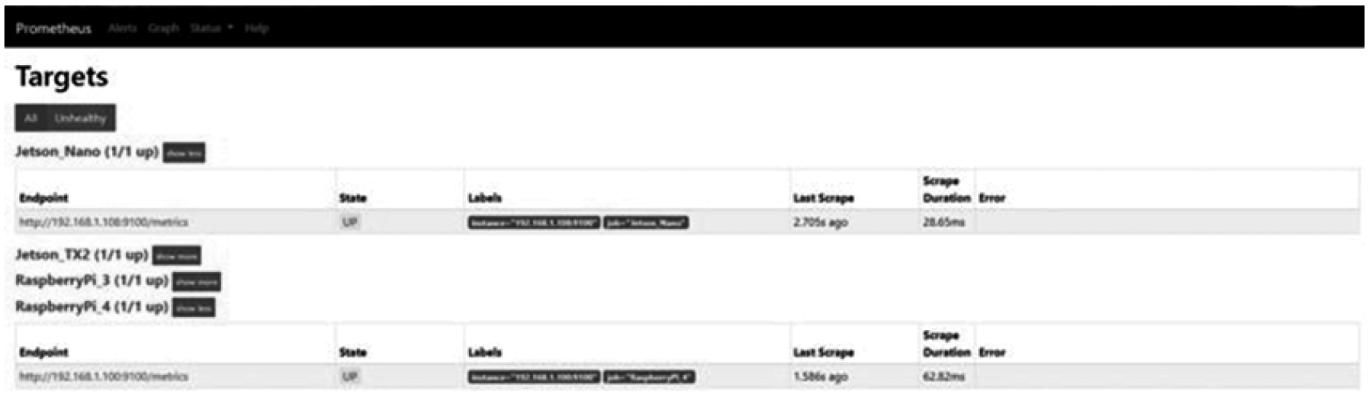
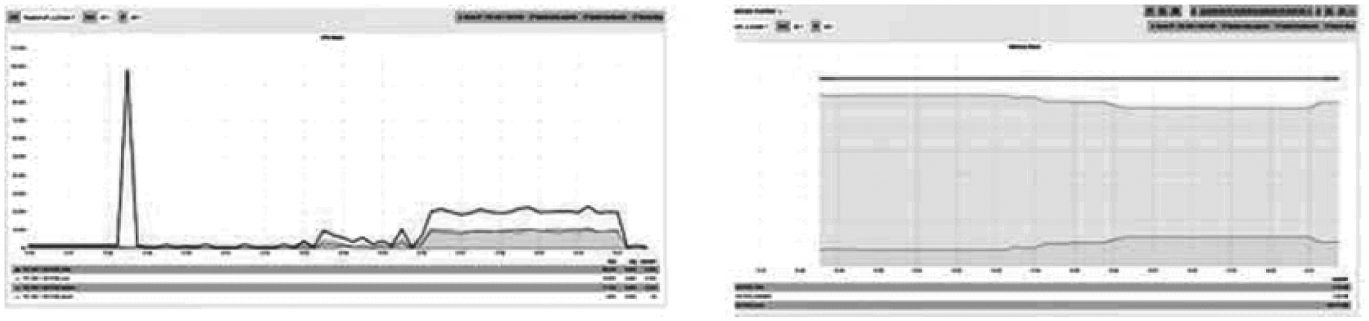


Рис. 3. Разнородные устройства, отслеживаемые на prometheus



(a) CPU (b) Memory

Рис. 4. Запуск на модели inception v3 FP 16

с помощью Docker container, которая обладает функциями контейнерных служб быстрого развертывания. В эксперименте с помощью Docker внедрение инструментов значительно повысило эффективность. Контейнеризация на основе услуг устраняет необходимость в сложные шаги, такие как ручная установка, и виртуализация контейнерной операционной системы создают архитектуру между разнородными устройствами. Кроме того, благодаря превосходному управлению контейнерами и кластерами в Kubernetes сервисные ошибки могут быть быстро перераспределены, что позволяет всей системе непрерывно отслеживать состояние всех устройств и эффективно расширять, и обновлять сервисные ресурсы.

ЛИТЕРАТУРА

1. Sukhija, Nitin, and Elizabeth Bautista. 2019. Towards a framework for monitoring and analyzing high performance computing environments using kubernetes and prometheus. In 2019 IEEE SmartWorld.https://doi.org/10.1109/SmartWorld-UIC-ATC-SCALCOM-IOPSCI.2019.00087
2. Marathe, Nikhil, Ankita Gandhi, and Jaimeel M. Shah. 2019. Docker swarm and kubernetes in cloud computing environment. In 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI). IEEE.
3. Chen, Wenyan, Kejiang Ye, and Cheng-Zhong Xu. 2019. Co-locating online workload and offline workload in the cloud: an interference analysis. In 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. https://doi.org/10.1109/HPCC/SmartCity/DSS.2019.00316
4. Medel, V., et al. 2018. Characterising resource management performance in Kubernetes. Computers & Electrical Engineering 68: 286–297. https://doi.org/10.1016/j.compeleceng.2018.03. 041.
5. Jiang, Congfeng, et al. 2020. Energy aware edge computing: A survey. Computer Communications. https://doi.org/10.1109/SEC.2018.00038.
6. Lingayat, Ashish, Ranjana R. Badre, and Anil Kumar Gupta. 2018. Performance evaluation for deploying docker containers on baremetal and virtual machine. In 2018 3rd International Conference on Communication and Electronics Systems (ICCES). IEEE. https://doi.org/10. 1109/CESYS.2018.8723998
7. Ning, Huansheng, et al. 2020. Heterogeneous edge computing open platforms and tools for internet of things. Future G
8. Agil Yolchuyev, «Extreme Gradient Boosting based Anomaly detection for Kubernetes Orchestration», 2023 27th International Conference on Information Technology (IT), pp.1–4, 2023.