Департамент образования и науки города Москвы

Государственное автономное образовательное учреждение высшего образования города Москвы

«Московский городской педагогический университет»

Институт цифрового образования

Департамент информатики, управления и технологий

ДИСЦИПЛИНА:

Интеграция и развертывание программного обеспечения с помощью контейнеров

**Лабораторная работа 1.1**

**Установка и настройка Docker. Работа с контейнерами в Docker**

Выполнил(а): Шведова С.С., группа: АДЭУ-211

Преподаватель: Босенко Т.М.

Москва

2025

**Цель работы**: освоить процесс установки и настройки Docker, научиться работать с контейнерами и образами Docker.

**Вариант 13.** Загрузить образ prometheus`, запустить контейнер, настроить конфигурацию Prometheus для сбора метрик и проверить доступность веб-интерфейса.

**Ход работы**

1. Установка Docker показана на рисунках 1-4.

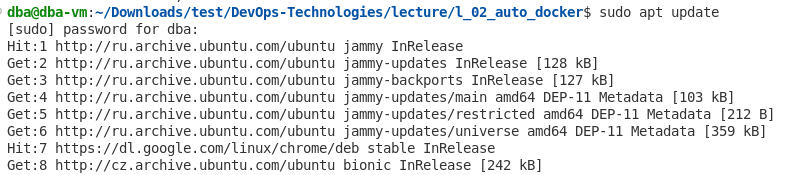


Рисунок 1. Установка обновлений

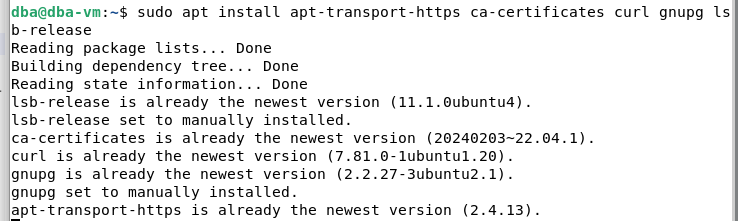


Рисунок 2. Установка четырех пакетов



Рисунок 3. Загрузка GPG-ключа, добавление нового репозитория Docker в систему

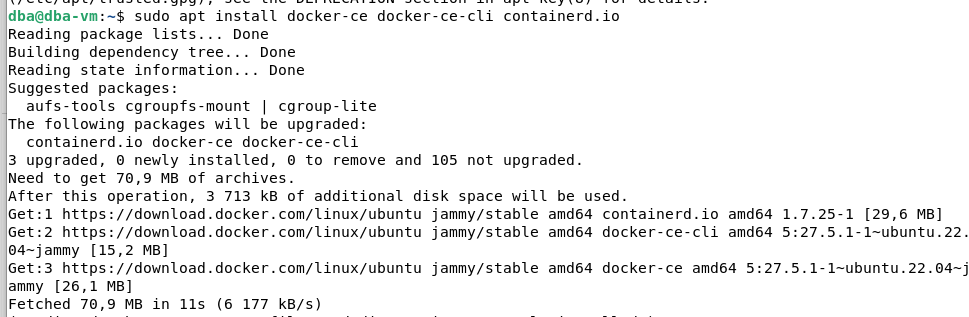


Рисунок 4. Установка трёх пакетов

Далее после завершения установки нужно добавить пользователя dba в группу docker, чтобы избежать необходимости использования sudo при выполнении команд Docker (рисунок 5). И также была выполнена команда на просмотр список запущенных контейнеров.

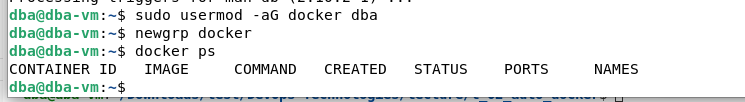


Рисунок 5. Добавление пользователя dba в группу docker

1. Проверка установки docker и знакомство с основными командами Docker CLI. На рисунке 6 выполнены проверки версии докера и его запуска (что он может загружать образы и запускать контейнеры). Еще также был выполнен просмотр всех контейнеров, включая остановленные.

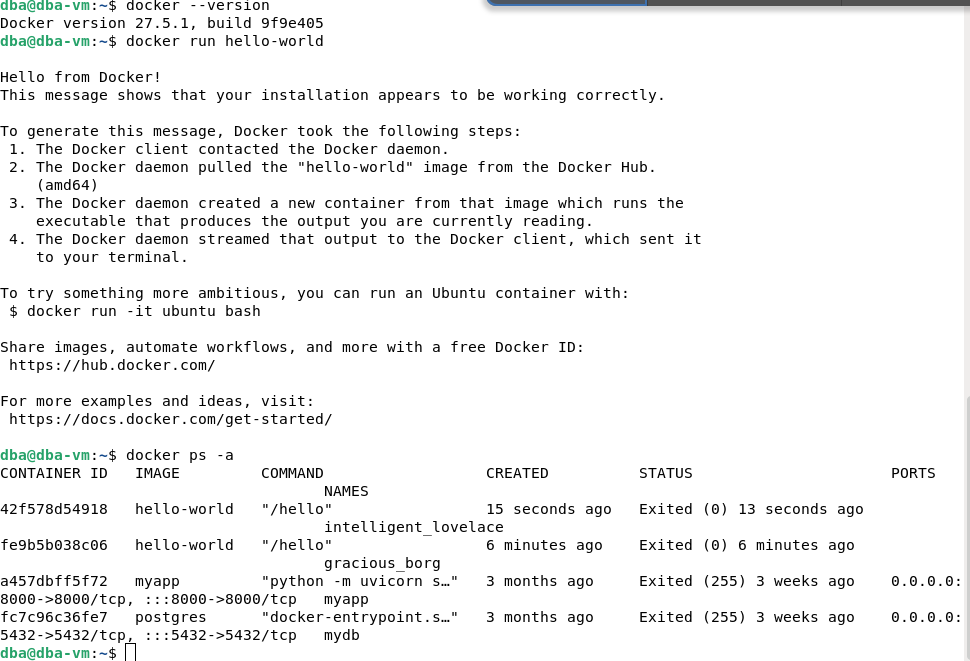


Рисунок 6. Проверка установки docker

На рисунке 7 показан просмотр список локальных образов Docker.

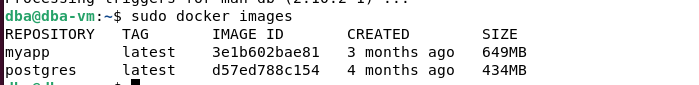


Рисунок 7. Просмотр список локальных образов Docker

1. Выполнение индивидуального задания. Сначала надо загрузить образ Prometheus и проверрить, что он загружен (рисунок 8).



Рисунок 8. Загрузка образа Prometheus

Затем нужно создать конфигурационный файл Prometheus с помощью команды nano ~/prometheus.yml (рисунок 9).

В этом конфигурационном файле определены следующие параметры:

1. global: раздел для определения глобальных параметров для всех jobs;
2. scrape\_interval: 15s: интервал сбора метрик по умолчанию — 15 секунд для всех jobs.
3. scrape\_configs: раздел, определяющий конкретные задачи (jobs) для сбора метрик.
4. job\_name: 'prometheus': имя задачи, указывающее, что это сбор метрик от Prometheus.
5. scrape\_interval: 5s: переопределяет глобальный интервал для этой задачи, устанавливая его на 5 секунд.
6. static\_configs: определяет статические цели для сбора метрик.
7. targets: ['localhost:9090']: указывает, что метрики будут собираться с локального сервера на порту 9090.

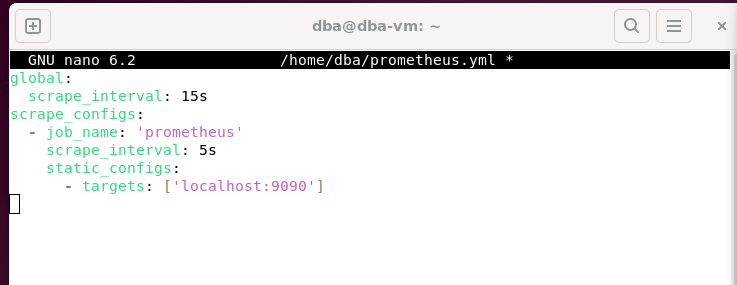


Рисунок 9. Создание конфигурационного файла

На рисунке 10 продемонстрирован запуск контейнера Prometheus с созданным конфигурационным файлом.



Рисунок 10. Запуск контейнера Prometheus с созданным конфигурационным файлом

Веб-интерфейс доступен по адресу http://localhost:9090. Как можно заметить, в интерфейсе в меню Status → Targets Prometheus собирает метрики (рисунок 11).

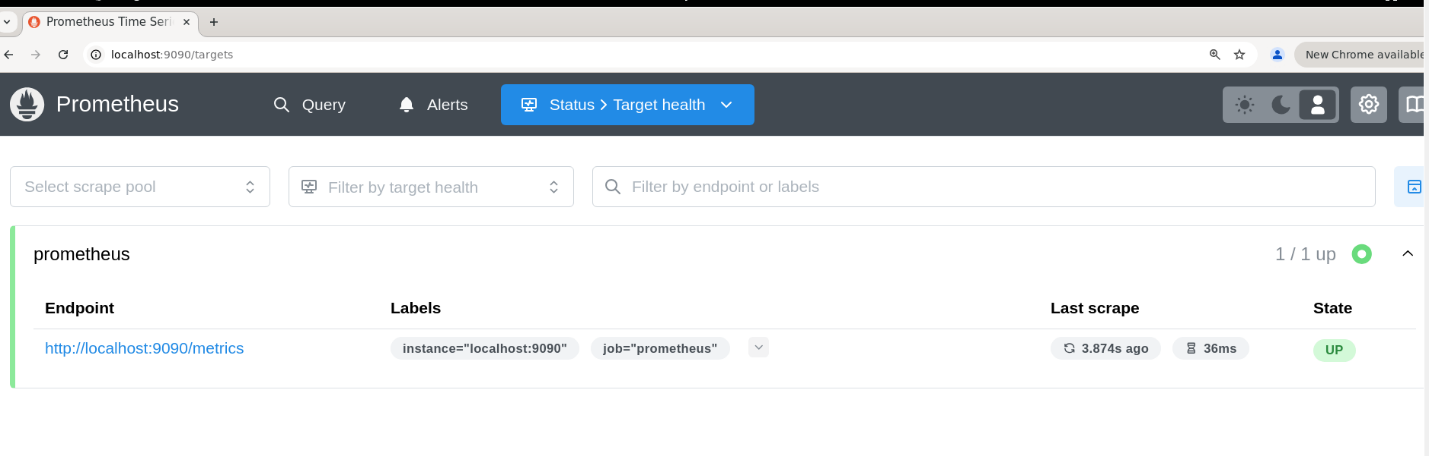


Рисунок 11. Веб-интерфейс работает и успешно собирает метрики

Остановка и удаление контейнера показано на рисунке 12.

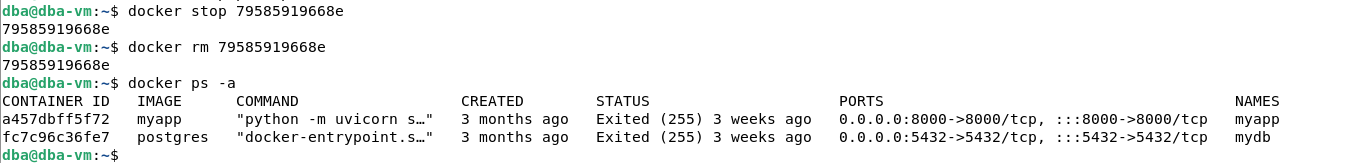


Рисунок 12. Остановка и удаление контейнера

**Выводы**

1. Установлен Docker на локальный компьютер.
2. Проверена корректность установки Docker.
3. Было выполнены основные командами Docker CLI для работы с образами и контейнерами.
4. Выполнено индивидуальное задание с загрузкой образа prometheus.

**Контрольные вопросы**

1. **Что такое Docker и для чего он используется?**

Docker — это платформа контейнеризации с открытым исходным кодом, используемая для оптимизации управления приложениями и разработки программного обеспечения. Используется для запуска одновременно нескольких рабочих процессов с меньшим потреблением ресурсов, чем обычно, для оптимизации и поддержки всего процесса разработки, в качестве экономичной альтернативы виртуальным машинам, как система контроля за приложениями и их разработкой

1. **Какие преимущества дает использование контейнеров Docker по сравнению с виртуальными машинами?**

* Изоляция. Контейнеры изолируют приложения друг от друга и от хост-системы, что уменьшает возможные конфликты между зависимостями и обеспечивает более надёжную среду выполнения.
* Портативность. Контейнеры могут быть созданы и запущены на различных системах без изменений, что делает развёртывание приложений более простым и надёжным.
* Эффективность использования ресурсов. Контейнеры используют общие ресурсы операционной системы и разделяют ядро, что делает их более эффективными по сравнению с виртуальными машинами.
* Масштабируемость. Контейнеры легко масштабируются горизонтально, позволяя быстро добавлять экземпляры приложения в ответ на увеличение нагрузки.
* Управление версиями. Контейнеры позволяют управлять версиями приложений и их зависимостей, что упрощает процесс развёртывания и обновления.

1. **Что такое образ Docker и как он связан с контейнерами?**

Docker-образ — это шаблон, из которого создаются Docker-контейнеры. Образ хранит в себе всё необходимое для запуска приложения, помещённого в контейнер: код, среду выполнения, библиотеки, переменные окружения и конфигурационные файлы. Образ связан с контейнерами: контейнер — это запущенная копия образа.

1. **Какие основные команды Docker CLI вы узнали в ходе выполнения лабораторной работы?**

* docker images, чтобы просмотреть список локальных образов Docker
* docker ps, чтобы просмотреть список запущенных контейнеров.
* docker ps -a, чтобы просмотреть список всех контейнеров, включая остановленные.

1. **Как можно настроить маршрутизацию портов при запуске контейнера Docker?**

Чтобы настроить маршрутизацию портов при запуске контейнера Docker, можно использовать сопоставление портов (маппинг). Оно позволяет перенаправлять запросы с порта хоста в контейнер и делать процессы внутри контейнера доступными извне.