

## **Тема и цель работы:**

Тема: docker

Цель работы: Научиться устанавливать, настраивать и использовать docker.

**Оборудование, ПО:**

Устройств о	Операционная система	IP адрес/Маска	Шлюз	DNS
CLI_A1	Windows server	192.168.1.1/24	-	-
CLI_A2	Astra Linux SE 1.7.x	192.168.1.2/24	192.168.1.1	-

## **Выполнение лабораторной работы:**

Docker — это одновременно платформа и технология для контейнеризации. Она позволяет создавать контейнеры и управлять ими для развёртывания и доставки кода на сервер.

Контейнер — это среда, внутри которой имитируется определённая операционная система. В эту систему мы можем положить код и запускать его в конкретной изолированной среде и в определённых нами условиях. Как правило, в одном контейнере запускают одно приложение или даже отдельный его компонент — модуль, функцию или микросервис.

Контейнер чем-то похож на виртуальную машину, только он гораздо компактнее и проще устроен. Он не требует выделять ему конкретные ресурсы, как виртуальная машина, а работает прямо на ресурсах нашей операционной системы.

Преимущества использования контейнеров Docker:

Возможность отката. В любой момент контейнер можно «сбросить», откатить до изначальной версии — со всем сервером так сделать не получится. Это позволяет эффективнее тестировать приложения и легче исправлять ошибки. Плюс контейнер всегда можно перезагрузить, как обычный компьютер, — иногда это тоже помогает исправить ошибку.

Готовая среда для запуска. Контейнер вместе с приложением сразу содержит среду для работы. В этой среде мы пишем приложение и в ней же его тестируем, чтобы убедиться в работоспособности. Это позволяет загрузить готовый контейнер на любой сервер — и быть уверенным, что приложение запустится нормально. Неважно, где и как код писали, запускаться он будет стабильно именно благодаря упаковке в контейнер.

Небольшой вес. Виртуальные машины могут весить несколько гигабайтов. Docker-контейнер чаще всего весит не больше пары сотен мегабайтов, иногда сильно меньше. Он быстро запускается и не требует больших вычислительных мощностей.

Изолированность от внешней среды. У docker-контейнера нет доступа к информации на хосте. Это делает работу приложения более безопасной.

Удобные инструменты управления. Образы docker хранятся в удобном docker-реестре, откуда можно в пару кликов запускать готовые среды и на их основе настраивать свои. Сам Docker позволяет управлять контейнерами: запускать, сохранять, редактировать, перезагружать. А если контейнеров в работе много, существуют специальные инструменты для массового управления — оркестрации. Для docker-контейнеров стандартным инструментом считается Kubernetes.

Основные компоненты Docker:

Docker состоит из четырёх основных компонентов: клиента, демона, хоста и реестра.

Клиент. Это наш инструмент, в который мы отдаём команды: поднять контейнер, настроить его, перезагрузить или удалить.

Демон. Это фоновый процесс, который управляет контейнерами. Он принимает команды от клиента и делает то, что его попросили. Плюс он всегда следит за контейнерами и в любой момент знает их состояние.

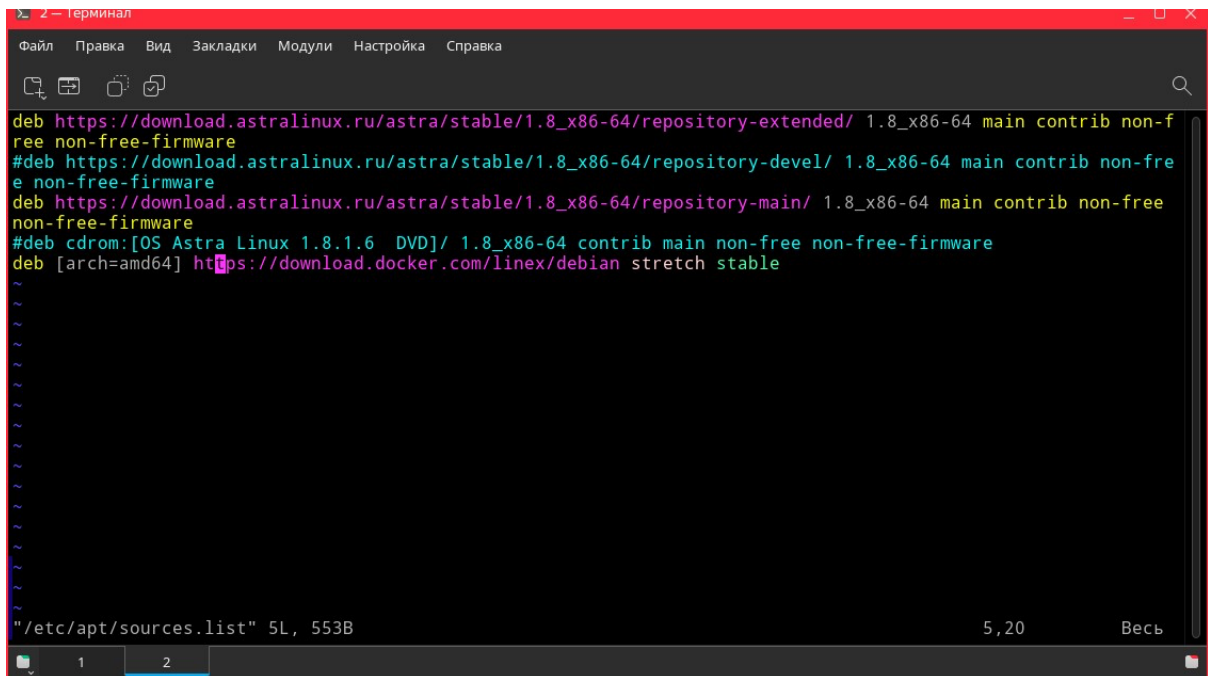
Хост. Тот сервер, на котором мы развернули Docker. Именно на нём в фоне работает демон, и на нём поднимаются контейнеры.

Реестр. Это наше хранилище образов, из которых демон разворачивает контейнеры. Это может быть общественный docker-хаб, в котором хранятся готовые образы. А может быть наше персональное хранилище, где лежат образы, сконструированные нами. Добавляем репозиторий, для этого открывает список репозитория

```
vim /etc/apt/sources.list
```

и вносим репозиторий

```
deb [arch=amd64] https://download.docker.com/linux/debian stretch stable
```



```
deb https://download.astralinux.ru/astra/stable/1.8_x86-64/repository-extended/ 1.8_x86-64 main contrib non-free non-free-firmware
#deb https://download.astralinux.ru/astra/stable/1.8_x86-64/repository-devel/ 1.8_x86-64 main contrib non-free non-free-firmware
deb https://download.astralinux.ru/astra/stable/1.8_x86-64/repository-main/ 1.8_x86-64 main contrib non-free non-free-firmware
#deb cdrom:[OS Astra Linux 1.8.1.6 DVD]/ 1.8_x86-64 contrib main non-free non-free-firmware
deb [arch=amd64] https://download.docker.com/linux/debian stretch stable
```

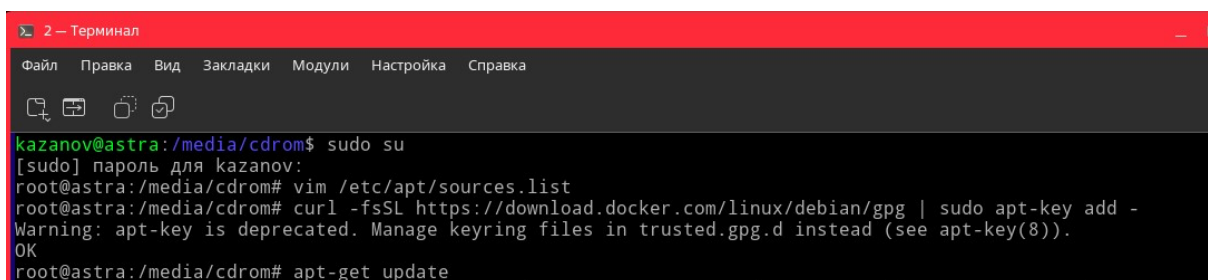
Рисунок 1-добавление ключа репозитория.

Добавляем ключ репозитория

```
curl -fsSL https://download.docker.com/linux/debian/gpg | sudo apt-key add -
```

Обновляем пакеты

```
apt-get update
```



```
kazanov@astra:/media/cdrom$ sudo su
[sudo] пароль для kazanov:
root@astra:/media/cdrom# vim /etc/apt/sources.list
root@astra:/media/cdrom# curl -fsSL https://download.docker.com/linux/debian/gpg | sudo apt-key add -
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
OK
root@astra:/media/cdrom# apt-get update
```

Рисунок 2-обновление apt-get.

Устанавливаем необходимые пакеты

```
sudo apt install docker.io
```

```
apt-get install apt-transport-https ca-certificates curl gnupg-agent
softwareproperties-common -y
```

```
root@astra:/media/cdrom# apt-get install docker-ce docker-ce-cli containerd.io -y
Чтение списков пакетов... Готово
Построение дерева зависимостей... Готово
Чтение информации о состоянии... Готово
```

Рисунок 3-установка необходимых пакетов.

```
root@astra:/media/cdrom# sudo apt install docker.io
Чтение списков пакетов... Готово
Построение дерева зависимостей... Готово
Чтение информации о состоянии... Готово
Уже установлен пакет docker.io самой новой версии (25.0.5.astra2+ci3).
```

Рисунок 4-установка других необходимых пакетов.

Добавим нашего пользователя в группу docker

usermod -aG docker user Перезапускаем Докер

systemctl restart docker

Проверяем статус Докера

systemctl status docker

```
root@astra:/media/cdrom# usermod -aG docker kazanov
root@astra:/media/cdrom# systemctl restart docker
root@astra:/media/cdrom# systemctl status docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/lib/systemd/system/docker.service; enabled; preset: enabled)
   Active: active (running) since Fri 2025-02-07 17:59:15 MSK; 3s ago
     TriggeredBy: ● docker.socket
        Docs: https://docs.docker.com
        Main PID: 24184 (dockerd)
```

Рисунок 5-проверка статуса активности.

Запускаем контейнер Portainer

docker run -d -p 9000:9000 -v /var/run/docker.sock:/var/run/docker.sock -v portainer\_data:/data portainer/portainer

```
root@astra:/media/cdrom# docker run -d -p 9000:9000 -v /var/run/docker.sock:/var/run/docker.sock -v portainer_data:/data portainer/portainer
76650e42cb63617aee103ac7a480bc31d1b7b10644f757fecca760959c47e383
```

Рисунок 6-запуск контейнера Portainer. После чего заходите на сетевой адрес вашего сервера на порт <http://localhost:9000/>

Вводим новый пароль для учетной записи Portainer "admin".

После чего попадаем в Portainer. Переходим на закладку "Local", после чего загрузится домашняя страница Docker'a.

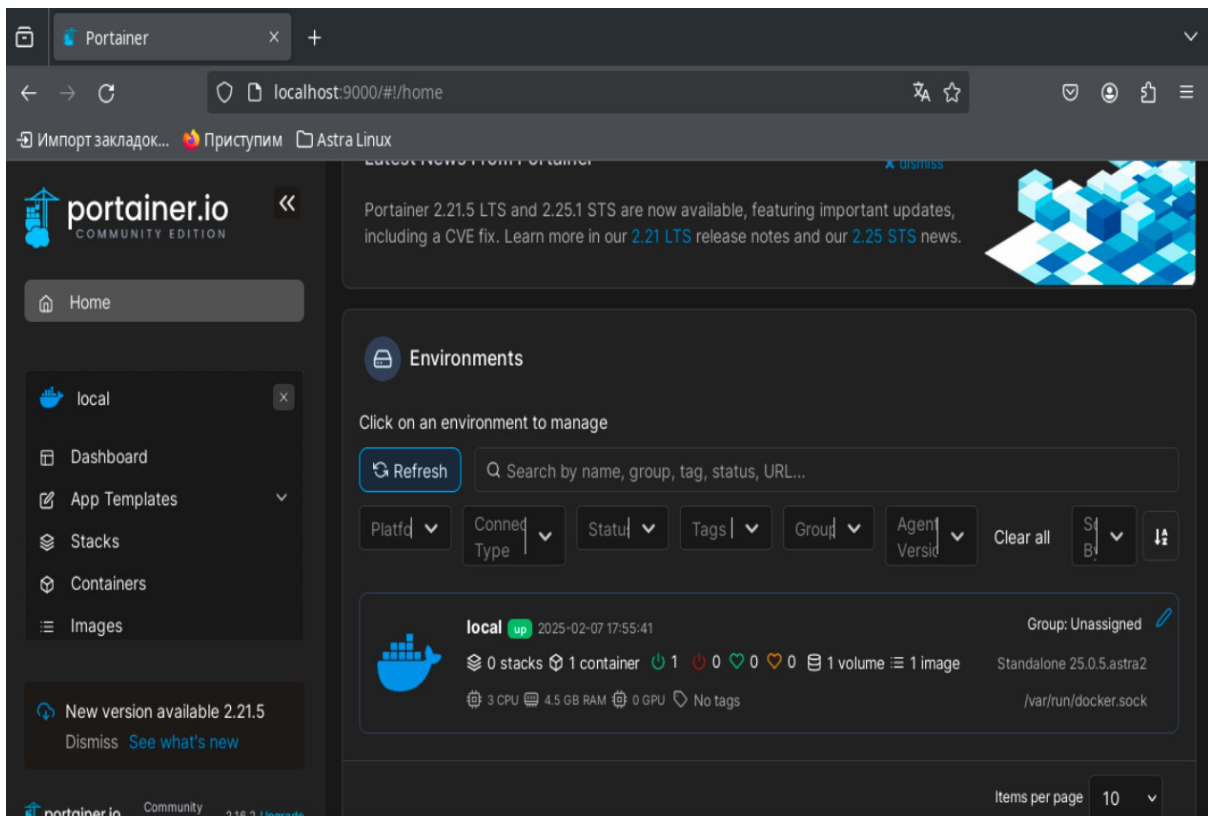


Рисунок 7-проверка сайта.

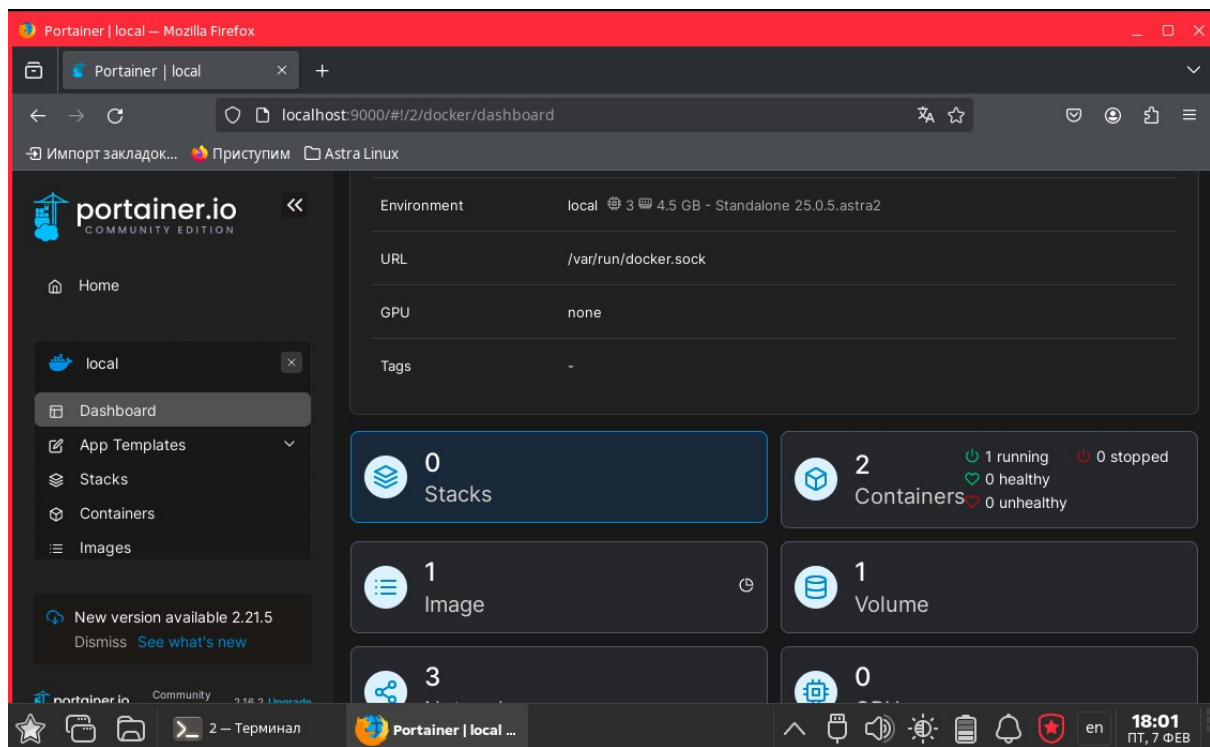


Рисунок 8-все работает без ошибок.



## **Вывод**

В ходе лабораторной работы я научился устанавливать и работать с docker.

## **Контрольные вопросы:**

### **1. Что такое докер?**

Docker — это платформа для контейнеризации, которая позволяет упаковывать, распространять и запускать приложения в изолированных средах, называемых контейнерами.

### **2. Что такое контейнер?**

Контейнер в docker это изолированное пространство, которое позволяет запускать приложения с их зависимостями от основной системы.