Информационные технологии (часть 2)

1. Требования к оформлению отчета

Отчет должен быть оформлен по СТП. Все рисунки должны быть читабельными. На каждом рисунке должна быть видна ФАМИЛИЯ СТУДЕНТА. Обязательно выделяем команды на рисунках в терминале!

2. Настройка ресурсов контейнеров docker (часть 2)

Введение

Целью данной лабораторной работы является изучение взаимодействия контейнеров docker с операционной системой.

Составить список используемых команд в терминале Linux

Команда	Значение	Расшифровать ключи

Создание общей папки для контейнера

Скачайте образ docker, содержащий средства разработки под linux:

```
sudo docker pull gcc
```

Запустите этот контейнер указав следующие ключи:

```
-it -v /home/osboxes/server:/home/osboxes
```

Ключ `-v` создает общую папку, доступную на виртуальной машине по пути `/home/osboxes/server`, а в контейнере по пути `/home/osboxes`.

На физическом компьютере или виртуальной машине создайте файл `hello.c` со следующим содержимым:

```
#include "stdio.h"
 main () {
     #if defined(_WIN32)
         printf("hello, windows\n");
     #elif defined(__linux__)
         printf("hello, linux\n");
     #elif defined(__APPLE__)
         printf("hello, Apple\n");
     #elif defined(BSD)
         printf("hello, BSD\n");
     #endif
 }
```

Скомпилируйте и запустите эту программу на физическом компьютере и в контейнере. Для запуска в контейнере нужно перейти в его терминале в общую папку, запустить компилятор `gcc`, и запустить результат компиляции командой `./a.out`.

Для групп 22зкм: скопируйте любую лабораторную работу из 1-го семестра по программированию в расшаренную папку, скомпилируйте ее в контейнере и запустите.

Проброс портов

Скачайте образ gitbucket (сервер системы контроля версий):

```
sudo docker pull gitbucket/gitbucket
```

При запуске контейнера ключ `-it` указывать не нужно, нужно указать ключи `-v` и `-p`.

Создайте на физической машине папку для хранения данных `gitbucket` и сделайте ее с помощью ключа `-v` общей с папкой контейнера `/gitbucket` (через общую папку на виртуальной машине).

Ключ `-p` нужен для проброса портов. Он имеет формат:

`-p` *host_port*`:`*container_port*

Сервер в контейнере запускается на порту 8080, этот порт следует указать в *container_port*. *host_port* – это порт, на котором будет доступен сервер на виртуальной машине (его нужно пробросить на физическую машину).

Дождитесь загрузки сервера (в терминале должен появиться текст `Server:main: Started`). В браузере компьютера зайдите на сервер `gitbucket`, аутентифицируйтесь под логином `root` с паролем `root`. Создайте репозиторий системы контроля версий с настройками по-умолчанию.

Перегрузите виртуальную машину, перезапустите контейнер и убедитесь, что учетная запись и репозиторий на сервере сохранились.

Взаимодействие контейнеров

По-умолчанию, контейнеры docker подключаются к виртуальной компьютерной сети через которую и осуществляется их взаимодействие. Каждый контейнер в этой сети имеет свой адрес.

В этом пункте мы будем с контейнера со средствами разработки выкладывать результаты на сервер контроля версий. Для этого нужно узнать его адрес (порт сервера уже известен – `8080`). Для этого:

- с помощью команды `sudo docker ps` определите имя контейнера с сервером.
- с помощью команды `sudo docker network inspect bridge` посмотрите конфигурацию виртуальной сети. В этой конфигурации в разделе `Containers` найдите контейнер с нужным именем. В параметрах этого контейнера будет примерно такая строчка `"IPv4Address": "172.17.0.3/16"`, где `172.17.0.3` – интересующий нас адрес контейнера сервера.

Далее создадим локальную копию репозитория и выложим ее на сервер. В терминале контейнера `gcc` выполните следующие действия:

- настройте пользователя системы контроля версий (обязательно указывать вашу эл. почту/имя):

```
git config --global user.email "you@example.com"

git config --global user.name "Your Name"
```

- создайте папку командой `mkdir` и перейдите в каталог для локального репозитория:

```
mkdir repo

cd repo
```

- командой `touch` создайте файл, а также создайте репозиторий и привяжите его к репозиторию на сервере:

```
touch LAB4.md

git init

git add LAB4.md

git commit -m "first commit"

git remote add origin http://server_ip:server_port/git/root/name.git

git push -u origin master
```

Вместо *name* укажите имя созданного ранее репозитория. Выполнение последней команды потребует ввода имени пользователя и пароля на сервере системы контроля версий. Используйте тот же логин, что и ранее `(root:root)`.

Перейдите в браузере на страницу репозитория и убедитесь, что в нем появился файл `LAB4.md`.