ПЕРМСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ

ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Факультет прикладной математики и механики

Кафедра вычислительной математики и механики

Реферат

Тема: “Программирование с использованием облачных платформ

Google Apps/Amazon/Azure”

Группа: ИТСИ-17-1м

Студент: Ермакова О.А.

Преподаватель: Истомин Д.А.

Пермь 2018

Содержание

[Часть 1 3](#_Toc535337514)

[1. Docker 3](#_Toc535337515)

[2. Amazon Simple Storage Service (S3) 5](#_Toc535337516)

[3. Google cloud platfom 7](#_Toc535337517)

[Часть 2 8](#_Toc535337518)

[1. Microsoft Azure: Mobile 8](#_Toc535337519)

# Часть 1 – Лабораторные работы

# Лабораторная работа №1 - Docker

Docker – это программная платформа для быстрой сборки, отладки и развертывания приложений. Docker упаковывает ПО в стандартизованные блоки, которые называются контейнерами. Каждый контейнер включает все необходимое для работы приложения: библиотеки, системные инструменты, код и среду исполнения. Благодаря Docker пользователи могут быстро развертывать и масштабировать свои приложения в любой среде и сохранять уверенность в том, что код будет работать.

В основе работы Docker лежит стандартизированный способ исполнения кода. Docker – это операционная система для контейнеров. Подобно тому как виртуальная машина создает виртуальное представление аппаратного обеспечения сервера (то есть устраняет необходимость непосредственно управлять таковым), контейнеры создают виртуальное представление серверной операционной системы. После установки на каждый сервер Docker предоставляет доступ к простым командам, необходимым для сборки, запуска или остановки контейнеров.

Использование Docker позволяет быстрее и эффективнее доставлять или перемещать код, стандартизирует выполняемые приложениями операции и в целом экономит средства, оптимизируя использование ресурсов. Благодаря Docker пользователи получают объект, который с высокой надежностью можно запускать на любой платформе. Простой синтаксис Docker предоставляет пользователям полный контроль над выполняемыми операциями. Повсеместное внедрение контейнеров подразумевает доступ к разнообразным инструментам и готовым приложениям, которые можно использовать с Docker.

Docker Swarm — это кластер докеров, множество хостов, на каждом из которых запущен отдельный Docker Engine, объединены под общим управлением, и выглядят как один большой докер. Хосты могут добавляться в кластер, и вычислительная ёмкость вашего большого докера будет тоже увеличена.

Docker Compose - это инструмент для определения и запуска многоконтейнерных приложений Docker. Для настройки служб приложения используется файл YAML.

HAProxy (High Availability Proxy) - популярный прокси сервер для Linux, Solaris и FreeBSD с возможностью балансировки нагрузки TCP/HTTP с открытым программным кодом. Его основная задача - повышение производительности серверной среды путем распределения рабочей нагрузки среди нескольких серверов (web, приложения, базы данных). Им пользуются такие известные проекты как GitHub, Imgur, Instagram и Twitter.

Алгоритм распределения нагрузки отвечает за выбор сервера для обработки запроса пользователя. HAProxy предлагает нам несколько алгоритмов. В дополнение к алгоритму, мы можем указать вес каждого сервера, в соответствии с которым он будет выбран чаще или реже.

* Roundrobin - алгоритм, применяемый по умолчанию. Перебирает сервера по очереди.
* Leastconn - выбирает сервер с наименьшим числом активных соединений. Рекомендуется к использованию для длинных сессий. Сервера в одном бекэнде варьируются циклически.
* Source - выбирает сервер исходя из хеша, построенного на основе IP пользователя. Таким образом, пользователь всегда обращается к одному и тому же серверу.

Алгоритм выполнения лабораторной работы:

1. Создание простого приложения на node.js с именем index.js:

var http = require('http');  
var os = require('os');

http.createServer(function (req, res) {  
 res.writeHead(200, {'Content-Type': 'text/html'});  
 res.end(`<h1>I'm ${[os.hostname()](https://millermedeiros.github.io/mdoc/examples/node_api/doc/os.html" \l "os.hostname" \t "_blank)}</h1>`);  
}).listen(8080);

1. Для упаковки приложения index.js создаем файл с именем Dockerfile:

FROM node  
RUN mkdir -p /usr/src/app  
COPY index.js /usr/src/app  
EXPOSE 8080  
CMD [ "node", "/usr/src/app/index" ]

1. Для создания образа Docker необходимо прописать в командной строке: docker build -t awesome.
2. Для балансировки нагрузки с помощью HAProxy необходимо создать контейнер с использованием инструмента Docker Compose. Для этого создаем файл docker-compose.yml со следующим содержимым:

version: '3'  
  
services:  
 awesome:  
 image: awesome  
 ports:  
 - 8080  
 environment:  
 - SERVICE\_PORTS=8080  
 deploy:  
 replicas: 20  
 update\_config:  
 parallelism: 5  
 delay: 10s  
 restart\_policy:  
 condition: on-failure  
 max\_attempts: 3  
 window: 120s  
 networks:  
 - web  
  
 proxy:  
 image: dockercloud/haproxy  
 depends\_on:  
 - awesome  
 environment:  
 - BALANCE=leastconn  
 volumes:  
 - /var/run/docker.sock:/var/run/docker.sock  
 ports:  
 - 80:80  
 networks:  
 - web  
 deploy:  
 placement:  
 constraints: [node.role == manager]  
  
networks:  
 web:  
 driver: overlay

1. Для создания Docker Swarm в командной строке прописываем: docker swarm init.
2. Для создания стека в командной строке прописываем: docker stack deploy --compose-file=docker-compose.yml prod .
3. Для запуска приложения в браузере открываем <http://localhost>.
4. Для просмотра всех реплик и сервисов используем команду: docker service ls.

# Лабораторная работа №2 - Amazon Simple Storage Service (S3)

Amazon Simple Storage Service (S3) — это долгосрочное хранилище данных на основе облачной обработки. Amazon S3 позволяет перемещать данные в облако и извлекать их обратно. В Amazon S3 хранятся не файлы, а объекты (objects), а объекты хранятся в "корзинах" (buckets). Объекты, хранимые в S3, не могут иметь размер, превышающий 5 Гбайт. Корзины (Buckets) существуют в плоском пространстве имен, предоставленном в общий доступ всем пользователям Amazon S3. А также buckets и object можно сделать доступными для публичного просмотра.

Доступ к S3

Прежде чем получить доступ к S3, необходимо создать учетную запись Amazon Web Services. Для этого на сайте <https://console.aws.amazon.com/iam/>. В левом боковом меню выбираем Пользователи -> Добавить пользователя. Получаем access\_key и secret\_access\_key. Также необходимо создать корзину (Bucket).

Алгоритм выполнения лабораторной работы:

1. Создание конфигурации для клиента:

AWSCredentials credentials = **new** BasicAWSCredentials(accessKey, secretKey);

ClientConfiguration clientConfig = **new** ClientConfiguration();

clientConfig.setProtocol(Protocol.HTTP);

AmazonS3 conn = **new** AmazonS3Client(credentials, clientConfig);

conn.setEndpoint("endpoint.com");

1. Создаем соединение для взаимодействия с сервером:

String accessKey = "insert your access key here!";

String secretKey = "insert your secret key here!";

AWSCredentials credentials = **new** BasicAWSCredentials(accessKey, secretKey);

AmazonS3 conn = **new** AmazonS3Client(credentials);

conn.setEndpoint("objects.dreamhost.com");

1. Загрузка объекта:

s3.putObject(PutObjectRequest.builder().bucket(bucket).key(key)

.build(),

RequestBody.fromByteBuffer(getRandomByteBuffer(10\_000)));

1. Скачивание объекта:

s3.getObject(GetObjectRequest.builder().bucket(bucket).key(key).build(),

ResponseTransformer.toFile(Paths.get("multiPartKey")));

1. Удаление объекта:

DeleteObjectRequest deleteObjectRequest = DeleteObjectRequest.builder().bucket(bucket).key(key).build();

s3.deleteObject(deleteObjectRequest);

# Лабораторная работа №3 - Google cloud platfom

Google Cloud Platform — предоставляемый компанией Google набор облачных служб, которые выполняются на той же самой инфраструктуре, которую Google использует для своих продуктов, предназначенных для конечных потребителей, таких как Google Search и YouTube. Кроме инструментов для управления, также предоставляется ряд модульных облачных служб, таких как облачные вычисления, хранение данных, анализ данных и машинное обучение. Для регистрации нужно иметь банковскую карту или банковский счет.

Google Cloud Platform предоставляет такие услуги, как инфраструктура как услуга, платформа как услуга, и бессерверные вычисления.

1. Создание Bukcets:

Bucket bucket =  
    storage.create(  
        BucketInfo.newBuilder(bucketName)  
            // See here for possible values: http://g.co/cloud/storage/docs/storage-classes  
            .setStorageClass(StorageClass.COLDLINE)  
            // Possible values: http://g.co/cloud/storage/docs/bucket-locations#location-mr  
            .setLocation("asia")  
            .build());

1. Загрузка объекта:

Storage storage = StorageOptions.getDefaultInstance().getService();  
BlobId blobId = BlobId.of("bucket", "blob\_name");  
BlobInfo blobInfo = BlobInfo.newBuilder(blobId).setContentType("text/plain").build();  
Blob blob = storage.create(blobInfo, "Hello, Cloud Storage!".getBytes(UTF\_8));

1. Удаление объекта:

BlobId blobId = BlobId.of(bucketName, blobName);  
boolean deleted = storage.delete(blobId);  
if (deleted) {  
  // the blob was deleted  
} else {  
  // the blob was not found  
}

# Часть 2 – Индивидуальная тема для изучения

# Microsoft Azure: Mobile

Служба Mobile Microsoft Azure предназначена для облегчения создания, развертывания и эксплуатации мобильных приложений для различных платформ.

Возможности:

* ∙ Быстрое создание мобильных приложений
* ∙ Быстрое добавление функции корпоративного единого входа
* ∙ Автономная синхронизация данных для построения быстро реагирующих при-
* ложений
* ∙ Подключение приложений к локальным данным
* ∙ Рассылка индивидуальных push-уведомлений
* ∙ Автоматическое масштабирование в соответствии с бизнес-требованиями

Важные функции для разработки мобильных приложений:

* Аутентификация и авторизация.
* Доступ к данным.
* Автономная синхронизация.
* Push-уведомления.
* Клиентские пакеты SDK.

Функции платформы, которые удобно использовать на рабочих сайтах для мобильных устройств:

* Автоматическое масштабирование.
* Промежуточные среды.
* Непрерывное развертывание.
* Виртуальные сети.
* Изолированные и выделенные среды.

**Цены на службу Mobile**

