МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

МАТЕРИАЛЫ ПРАКТИЧЕСКИХ ЗАНЯТИЙ

Технологии хранения в системах кибербезопасности

(наименование дисциплины (модуля) в соответствии с учебным планом)

Уровень	специалитет
	(бакалавриат, магистратура, специалитет)
Форма обучения	очная
	(очная, очно-заочная, заочная)
Направление(-я) подготовки	10.05.04 Информационно-аналитические системы безопасности
	(код(-ы) и наименование(-я))
Институт	Кибербезопасности и цифровых технологий (ИКБ)
	(полное и краткое наименование)
Кафедра	КБ-2 «Информационно-аналитические системы кибербезопасности»
	(полное и краткое наименование кафедры, реализующей дисциплину (модуль))
Лектор	к.т.н., Селин Андрей Александрович, Бугаев Александр Александрович
	(сокращенно – ученая степень, ученое звание; полностью – ФИО)

Используются в данной редакции с учебного года 2024/2025

(учебный год цифрами)

Проверено и согласовано «____» _____2024 г. _____ А.А. Бакаев

(подпись директора Института/Филиала с расшифровкой)

Москва 2024 г.

ПРАКТИЧЕСКАЯ РАБОТА № 8

«Знакомство с фреймворком распределенной обработки данных Apache Spark»

Цель работы – получение практических навыков использования фреймворка Apache Spark для обработки и анализа данных.

Задание:

1. Запустите Unix-подобную систему (например, Debian 12.6.0 64-bit[1]).

2. Создайте пользователя с именем формата **fio_nn**,
где f – первая буква фамилии на латинице;
i – первая буква имени на латинице;
o – первая буква отчества на латинице (при наличии),
nn – двузначный номер по списку в группе.

Добавьте его в группу sudo. **Все дальнейшие действия необходимо выполнять от имени созданного пользователя.**

3. Запустите терминал и установите Docker и Docker Compose.

4. Создайте каталог для нового проекта (например, spark) и сформируйте файл **docker-compose.yml** для развертывания контейнера с **Jupyter Notebook** и **PySpark** (https://quay.io/repository/jupyter/pyspark-notebook).

Требования к запускаемым сервисам:

– последние 2 цифры номера порта, на котором будет развернут сервис, должны соответствовать номеру по списку в группе (например, для 15 – 12315, 8015, 9915 и т.п.);

– имя контейнера должно заканчиваться на символ подчеркивания и инициалы ФИО (например, для Иванова Петра Дмитриевича – pyspark_ipd).

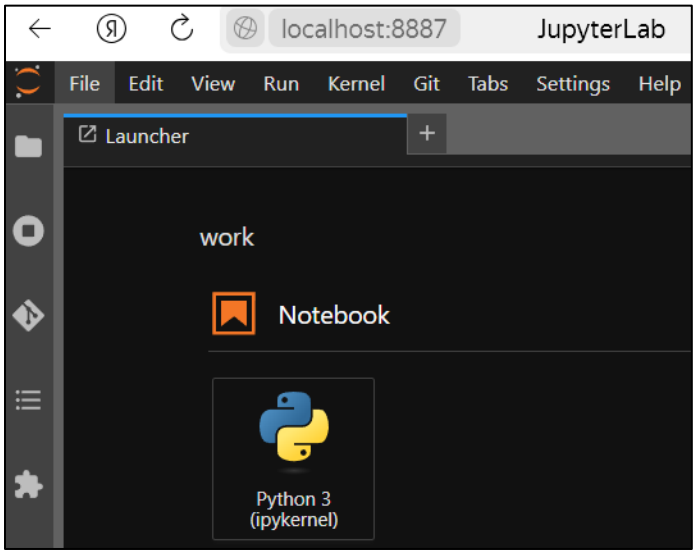
[1] Можно скачать готовый образ виртуальной машины по ссылке https://sourceforge.net/projects/osboxes/files/v/vb/14-D-b/12.6.0/64bit.7z/download

Пример файла docker-compose.yml:

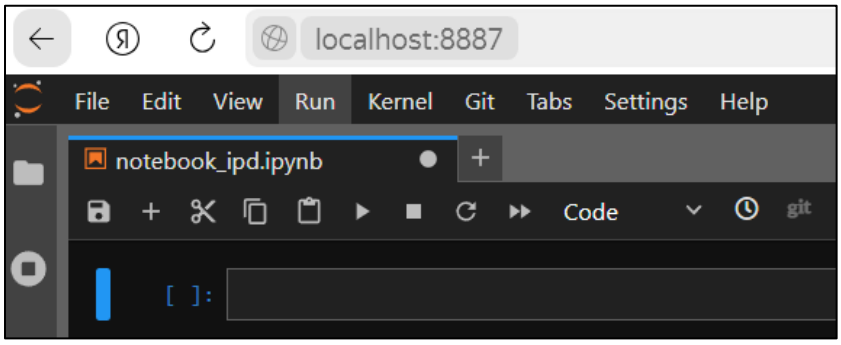
```yaml
services:
  pyspark:
    image: quay.io/jupyter/pyspark-notebook:latest
    container_name: pyspark_ipd
    volumes:
      - ./notebooks:/home/jovyan/work
    ports:
      - 8887:8888
      - 4047:4040
      - 4147:4041
    networks:
      - pyspark-net

networks:
  pyspark-net:
    driver: bridge
```

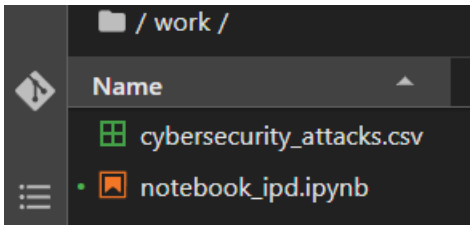
5. Разверните контейнер с помощью Docker Compose.



6. Создайте блокнот (notebook). Название блокнота должно заканчиваться на символ подчеркивания и инициалы ФИО.



7. Загрузите CSV-файл **cybersecurity_attacks.csv** (https://github.com/incribo-inc/cybersecurity_attacks), содержащий информацию о кибератаках, и поместите его в свой рабочий каталог.



8. Изучите документацию Spark SQL API: https://spark.apache.org/docs/latest/api/python/reference/pyspark.sql/index.html.
При выполнении последующих заданий обращайтесь к ней.

9. Импортируйте модули библиотеки PySpark, содержащие функции и типы данных для работы с данными в Spark DataFrame (pyspark.sql.functions, pyspark.sql.types). Создайте сессию Spark. Название приложения должно заканчиваться на символ подчеркивания и инициалы ФИО.

```
from pyspark.sql import SparkSession
import pyspark.sql.functions as f
from pyspark.sql.types import StructType, StructField, IntegerType, StringType, DoubleType, DateType

# Создание сессии Spark
spark = SparkSession.builder.appName("spark_ipd").getOrCreate()
```

Во всех блоках кода должен быть комментарий, поясняющий, что он выполняет. Все действия необходимо выполнять с помощью библиотеки PySpark. Демонстрация кода и полученных результатов обязательна.

10. Сформируйте DataFrame (название должно заканчиваться на символ подчеркивания и инициалы ФИО), загрузив данные из файла cybersecurity_attacks.csv с помощью функции **spark.read.csv**.

```
df_ipd = spark.read.csv("cybersecurity_attacks.csv", header=True, multiLine=True, inferSchema=True)
```

Проверьте схему полученных данных с помощью функции **printSchema**. **Красным выделен минимальный набор столбцов для выполнения последующих заданий.**

```
df_ipd.printSchema()
```

```
root
 |-- Timestamp: timestamp (nullable = true)
 |-- Source IP Address: string (nullable = true)
 |-- Destination IP Address: string (nullable = true)
 |-- Source Port: integer (nullable = true)
 |-- Destination Port: integer (nullable = true)
 |-- Protocol: string (nullable = true)
 |-- Packet Length: integer (nullable = true)
 |-- Packet Type: string (nullable = true)
 |-- Traffic Type: string (nullable = true)
 |-- Payload Data: string (nullable = true)
 |-- Malware Indicators: string (nullable = true)
 |-- Anomaly Scores: double (nullable = true)
 |-- Alerts/Warnings: string (nullable = true)
 |-- Attack Type: string (nullable = true)
 |-- Attack Signature: string (nullable = true)
 |-- Action Taken: string (nullable = true)
 |-- Severity Level: string (nullable = true)
 |-- User Information: string (nullable = true)
 |-- Device Information: string (nullable = true)
 |-- Network Segment: string (nullable = true)
 |-- Geo-location Data: string (nullable = true)
 |-- Proxy Information: string (nullable = true)
 |-- Firewall Logs: string (nullable = true)
 |-- IDS/IPS Alerts: string (nullable = true)
 |-- Log Source: string (nullable = true)
```

Сделать выборку необходимых полей можно с помощью функции **select**.

```
selected_columns = ['Timestamp', 'Source IP Address', 'Destination IP Address']
df_ipd = df_ipd.select(selected_columns)
df_ipd.show(5)
```

```
+-------------------+-----------------+----------------------+
|          Timestamp|Source IP Address|Destination IP Address|
+-------------------+-----------------+----------------------+
|2023-05-30 06:33:58|    103.216.15.12|          84.9.164.252|
|2020-08-26 07:08:30|   78.199.217.198|        66.191.137.154|
|2022-11-13 08:23:25|     63.79.210.48|         198.219.82.17|
|2023-07-02 10:38:46|    163.42.196.10|       101.228.192.255|
|2023-07-16 13:11:07|    71.166.185.76|       189.243.174.238|
+-------------------+-----------------+----------------------+
only showing top 5 rows
```

Приведите поля к соответствующим типам данных (можно делать по мере необходимости) одним из 2 способов:

– предварительно создав схему данных и использовав ее при загрузке (**на рисунке схема неполная**);

```
schema = StructType([
    StructField("Timestamp", DateType(), True),
    StructField("Source IP Address", StringType(), True),
    StructField("Destination IP Address", StringType(), True),
    StructField("Source Port", IntegerType(), True),
    StructField("Destination Port", IntegerType(), True)
])

df_ipd = spark.read.csv("cybersecurity_attacks.csv", header=True, multiLine=True, schema=schema)
df_ipd.printSchema()
```

```
root
 |-- Timestamp: date (nullable = true)
 |-- Source IP Address: string (nullable = true)
 |-- Destination IP Address: string (nullable = true)
 |-- Source Port: integer (nullable = true)
 |-- Destination Port: integer (nullable = true)
```

– используя функцию преобразования типа данных в столбце.

```
df_ipd = df_ipd.withColumn("Timestamp", df_ipd["Timestamp"].cast(DateType()))
```

11. Выведите 10 записей, в которых порт источника (Source Port) делится без остатка на ваш номер по списку, умноженный на 10.

12. Выведите результат группировки по типу трафика (Traffic Type) с подсчетом количества записей.

13. Выведите результат группировки по протоколу (Protocol) с подсчетом среднего размера пакета.

14. Выведите 10 IP-адресов получателя, отсортированных в порядке убывания.

15. Сформируйте свой тестовый набор данных (не менее 7 столбцов и не менее 10000 записей). Можно использовать сервис https://sqldatagenerator.com/generator.

16. Проведите не менее 5 манипуляций с новым набором данных с помощью PySpark (с пояснениями и демонстрацией).

17. **Дополнительно**: изучите возможности построения диаграмм с помощью PySpark и библиотеки matplotlib.