Chapter 3: Spark RDDs

Ex1: KDD Cup 99

In [1]:
```
# link dataset: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html
```

Cho dữ liệu KDDCup99 (xem thông tin chi tiết ở link trên và tải dữ liệu kddcup.data_10_percent.gz về để làm bài)

Yêu cầu:

1. Đọc dữ liệu từ file "kddcup.data_10_percent.gz". Tạo RDD từ dữ liệu này
2. Cho biết dữ liệu có bao element.
3. In thông tin 3 element đầu tiên của RDD
4. Tạo một RDD mới từ RDD trên, với điều kiện các element trong RDD mới đều có chuỗi "normal."
5. Cho biết RDD mới có bao nhiêu element, mất bao nhiêu thời gian để đếm số element này?
6. Tạo array chứa tất cả các element của RDD ban đầu, mất bao nhiêu thời gian để thực hiện việc này?
7. Tạo một RDD sample bằng 1/10 RDD ban đầu, các element lấy ngẫu nhiên.
8. Cho biết tỉ lệ các element chứa chuỗi "normal." so với tổng số element trong RDD sample, mất bao nhiêu thời gian để thực hiện công việc này?
9. Tạo RDD sample với các element không chứa từ "normal." từ RDD sample trên. Số element của RDD này là bao nhiêu? In thông tin element đầu tiên của RDD này.
10. Với RDD ban đầu, mỗi element là một chuỗi. Hãy tạo ra một RDD mới (array_RDD) với mỗi element là một danh sách các phần tử được tách ra từ chuỗi với dấu phân cách là ",". In thông tin element đầu tiên của RDD này.
11. Với array_RDD trên, hãy cho biết có những protocol nào được sử dụng? Có bao nhiêu protocol? (biết protocol là phần tử thứ 2 trong từng element)
12. Với array_RDD trên, hãy cho biết có những service nào được sử dụng? Có bao nhiêu service? (biết service là phần tử thứ 3 trong từng element)
13. Một sản phẩm được tạo ra sẽ có một protocol và một service, tạo một bảng danh sách các sản phẩm bằng cách phối hợp từng protocol và service của 2 câu trên (gợi ý: dùng RDD1.cartesian(RDD2).collect(). In kết quả. Có bao nhiêu sản phẩm được tạo ra?
14. Cho biết số lượng partition của array_RDD.
15. Lưu array_RDD vào thư mục "kdd_cup" với mỗi partition là 1 file riêng lẻ.

In [2]:
```
import findspark
findspark.init()
```

In [3]:
```
import pyspark
from pyspark import SparkContext
from pyspark.sql import SparkSession
```

In [4]:
```
sc = SparkContext()
```

In [5]:
```
#1.
file = "kddcup.data_10_percent.gz"
data = sc.textFile(file)
```

In [6]:
```
#2.
print("Lines:", data.count())
```

```
Lines: 494021
```

In [7]:
```
#3.
data.take(3)
```

Out[7]:
```
['0,tcp,http,SF,181,5450,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,8,8,0.00,0.00,0.00,0.0
0,1.00,0.00,0.00,9,9,1.00,0.00,0.11,0.00,0.00,0.00,0.00,0.00,normal.',
 '0,tcp,http,SF,239,486,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,8,8,0.00,0.00,0.00,0.0
0,1.00,0.00,0.00,19,19,1.00,0.00,0.05,0.00,0.00,0.00,0.00,0.00,normal.',
 '0,tcp,http,SF,235,1337,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,8,8,0.00,0.00,0.00,0.0
0,1.00,0.00,0.00,29,29,1.00,0.00,0.03,0.00,0.00,0.00,0.00,0.00,normal.']
```

In [8]:
```
data.first()
```

Out[8]:
```
'0,tcp,http,SF,181,5450,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,8,8,0.00,0.00,0.00,0.0
0,1.00,0.00,0.00,9,9,1.00,0.00,0.11,0.00,0.00,0.00,0.00,0.00,normal.'
```

In [9]:
```
#4.
normal_data = data.filter(lambda x: 'normal.' in x)
```

In [10]:
```
#5.
from time import time
t0 = time()
normal_count = normal_data.count()
t1 = time() - t0
print("There are", normal_count, "'normal's")
print("Time to count:", format(round(t1,3)), "seconds")
```

```
There are 97278 'normal's
Time to count: 1.351 seconds
```

In [11]:
```
#6. Array
t0 = time()
array_data = data.collect()
t1 = time() - t0
print("Data collected in", round(t1,3), "seconds")
```

```
Data collected in 3.126 seconds
```

In [12]:
```
#7.
data_sample = data.sample(False, 0.1, 42)
# parameter 1: the sampling is done with replacement or not
# parameter 2: the sample size as a fraction.
# parameter 3: [optionally] provide a random seed.
sample_size = data_sample.count()
total_size = data.count()
print("Sample size is", sample_size, "of", total_size)
```

```
Sample size is 49387 of 494021
```

In [13]:
```
#8.
sample_normal_tags = data_sample.filter(lambda x: "normal." in x)

t0 = time()
sample_normal_tags_count = sample_normal_tags.count()
tt = time() - t0

sample_normal_ratio = sample_normal_tags_count / float(sample_size)

print("Sample with normal size is", sample_normal_tags_count)
print("The ratio of 'normal' interactions is ", round(sample_normal_ratio,3))
print("Count done in", round(tt,3) ,"seconds")
```

```
Sample with normal size is 9733
The ratio of 'normal' interactions is  0.197
Count done in 1.397 seconds
```

In [14]:
```
#9.
sample_without_normal = data_sample.subtract(sample_normal_tags)
```

In [15]:
```
print("Sample without normal size is", sample_without_normal.count())
```

```
Sample without normal size is 39654
```

In [16]:
```
sample_without_normal.take(1)
```

Out[16]:
```
['0,icmp,ecr_i,SF,1032,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,509,509,0.00,0.00,0.0
0,0.00,1.00,0.00,0.00,255,255,1.00,0.00,1.00,0.00,0.00,0.00,0.00,0.00,smurf.']
```

In [17]:
```
#10. array_RDD
array_RDD = data.map(lambda x: x.split(","))
```

In [18]:
```
array_RDD.first()
```

Out[18]:
```
['0',
 'tcp',
 'http',
 'SF',
 '181',
 '5450',
 '0',
 '0',
 '0',
 '0',
 '0',
 '1',
 '0',
 '0',
 '0',
 '0',
 '0',
 '0',
 '0',
 '0',
 '0',
 '0',
 '8',
 '8',
 '0.00',
 '0.00',
 '0.00',
 '0.00',
 '1.00',
 '0.00',
 '0.00',
 '9',
 '9',
 '1.00',
 '0.00',
 '0.11',
 '0.00',
 '0.00',
 '0.00',
 '0.00',
 '0.00',
 'normal.']
```

In [19]:
```
#11.
protocols = array_RDD.map(lambda x: x[1]).distinct()
protocols.collect()
```

Out[19]:
```
['tcp', 'udp', 'icmp']
```

In [20]:
```
print("Protocol:", protocols.count())
```

```
Protocol: 3
```

In [21]:
```
#12
services = array_RDD.map(lambda x: x[2]).distinct()
services.collect()
```

Out[21]:
```
['http',
 'smtp',
 'finger',
 'domain_u',
 'auth',
 'telnet',
 'ftp',
 'eco_i',
 'ntp_u',
 'ecr_i',
 'other',
 'private',
 'pop_3',
 'ftp_data',
 'rje',
 'time',
 'mtp',
 'link',
 'remote_job',
 'gopher',
 'ssh',
 'name',
 'whois',
 'domain',
 'login',
 'imap4',
 'daytime',
 'ctf',
 'nntp',
 'shell',
 'IRC',
 'nnsp',
 'http_443',
 'exec',
 'printer',
 'efs',
 'courier',
 'uucp',
 'klogin',
 'kshell',
 'echo',
 'discard',
 'systat',
 'supdup',
 'iso_tsap',
 'hostnames',
 'csnet_ns',
 'pop_2',
 'sunrpc',
 'uucp_path',
 'netbios_ns',
 'netbios_ssn',
```

```
 'netbios_dgm',
 'sql_net',
 'vmnet',
 'bgp',
 'Z39_50',
 'ldap',
 'netstat',
 'urh_i',
 'X11',
 'urp_i',
 'pm_dump',
 'tftp_u',
 'tim_i',
 'red_i']
```

In [22]:
```
print("Service:", services.count())
```

```
Service: 66
```

In [23]:
```
#13
product = protocols.cartesian(services).collect()
```

In [24]:
```
product
```

Out[24]:
```
[('tcp', 'http'),
 ('tcp', 'smtp'),
 ('tcp', 'finger'),
 ('tcp', 'domain_u'),
 ('tcp', 'auth'),
 ('tcp', 'telnet'),
 ('tcp', 'ftp'),
 ('tcp', 'eco_i'),
 ('tcp', 'ntp_u'),
 ('tcp', 'ecr_i'),
 ('tcp', 'other'),
 ('tcp', 'private'),
 ('tcp', 'pop_3'),
 ('tcp', 'ftp_data'),
 ('tcp', 'rje'),
 ('tcp', 'time'),
 ('tcp', 'mtp'),
 ('tcp', 'link'),
 ('tcp', 'remote_job'),
```

In [25]:
```
print("There are", len(product), "combinations of protocol X service")
```

```
There are 198 combinations of protocol X service
```

In [26]:
```
#14.
array_RDD.getNumPartitions()
```

Out[26]:
```
1
```

In [28]:
```
#15.
array_RDD.saveAsTextFile("kdd_cup")
```