

Apache Server Log Analysis.

Description

The Apache services such as Hadoop, Spark, Tomcat, and Hive run on most data engineering servers throughout the world. All the services follow the same pattern because all are open source. You are a data engineer who works for a start-up named "Hadoop Analytics," which serves major clientele.

You have been assigned to one of their prestigious clients to resolve a production issue. As you are dealing with Hadoop, you are familiar with the working of logs. The server's information is stored in the logs along with the information listed below:

1. Resource details
2. Identification of the person who accessed the logs
3. Date and time the logs were accessed
4. Specifications on any problems that emerge
5. Information about the final product

Objective:

Perform server log analysis to assist businesses in identifying and analyzing critical business errors, as well as potential customers and their domains

Dataset Name: server-access-log.txt (Can be downloaded from the Course Resources tab)

Access_log file description:

Snippet:


```
10.1.2.3 - reh [10/Nov/2021:19:22:12 -0000] "GET /sematext.png HTTP/1.1" 200 3423
```

The following elements are present in the dataset:

1. **%h**: Resolved into **10.1.2.3** – the IP address of the remote host that made the request
2. **%l**: Identd provides the remote log name, with a hyphen, which is a value that can be logged if the information provided by the logging directive cannot be located or accessed
3. **%u**: Resolved into **rehg**, the user identifier determined by the HTTP authentication
4. **%t**: The date and time of the request with the time zone; in the above case it is **[10/Nov/2021:19:22:12 -0000]**
5. **%r**: The first line of the request inside double quotes; in the above case it is **"GET /sematext.png HTTP/1.1"**
6. **%s**: The status code reported to the client
This information is crucial because it determines whether the request was successful or not.
7. **%b**: The size of the object sent to the client; in our case, the object was the **sematext.png** file and its size was **3423** bytes.

Step 1: Upload the “server-access-log” file to the HDFS

```
hdfs dfs -put server-access /user/swapnasamirshukla1988gmail/logs_dataset
```

	Name	Size	Date	Time
<input type="checkbox"/>	 data-files		11/12/22	
<input type="checkbox"/>	 insta-cart		15/12/22	
<input type="checkbox"/>	server-access		12/02/24	10:49

```
s1-cdp-prod-en21 login: swapnasamirshukla1988gmai
Password:
Last login: Sun Feb  4 18:43:35 on pts/27

[swapnasamirshukla1988gmai@s1-cdp-prod-en21 ~]$ cat /dev/urandom | tr -dc 'a-z0-9' | fold -w 64 | xargs echo
=====
*                               :

Password for swapnasamirshukla1988gmai@CDP-ENV.GNE4-RUTX.CLOUDERA.SITE:
[swapnasamirshukla1988gmai@s1-cdp-prod-en21 ~]$ hdfs dfs -put server-access /user/swapnasamirshukla1988gmai/logs_dataset
[swapnasamirshukla1988gmai@s1-cdp-prod-en21 ~]~$
```

Step 2:Execute the PySpark Commands following the below steps:

- Login to the web console
- To enter the PySpark console run the following command:

Command: pyspark3

```
[swapasamirshukla@sl-cdp-prod-en21 ~]$ pyspark3
Python 3.6.8 (default, Nov 16 2020, 16:55:22)
[GCC 4.8.5 20150623 (Red Hat 4.8.5-44)] on linux
Type "help", "copyright", "credits" or "license()" for more information.
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
24/02/04 15:08:35 WARN util.Utils: Service 'SparkUI' could not bind on port 4040. Attempting port 4041.
/opt/cloudera/parcels/CDH-7.2.15-1.cdh7.2.15.p1.26792553/lib/spark3/python/pyspark/context.py:238: FutureWarning: Python 3.6 support is deprecated in Spark 3.2.
FutureWarning
Welcome to

      ____
     / __ \
    / ___/
   /  __ \
  /___/  \
         version 3.2.1.7.2.15.1-1


Using Python version 3.6.8 (Nov 16 2020 16:55:22)
Spark context Web UI available at http://sl-cdp-prod-en21.cdp-env.gne4-rutx.cloudera.site:4041
Spark context available as 'sc' (master = local[*], app id = local-1707059315807).
SparkSession available as 'spark'.

>>>
```

Step 3: Perform the below tasks on the uploaded dataset:

- **Status code analysis:**
 - Read the log file as an RDD in PySpark
 - Consider the sixth element as it is “request type” and replace the “single quote" with blank
 - Convert each word into a tuple of (word,1)
 - Apply “reduceByKey“ transformation to count the values
 - Display the data

#Read the log file into an RDD

```
log_rdd = spark.sparkContext.textFile("logs_dataset/server-access-log.txt")
```

Take the first 10 lines of the RDD

```
contents = log_rdd.take(10)
```

Print each of these lines

for line in contents:

```
    print(line)
```

Extract the status code and map each to a tuple

```
status_code_rdd = log_rdd.map(lambda line: (line.split(" ")[5].replace("'", ""), 1))
```

Reduce by key to count occurrences of each status code

```
status_code_counts = status_code_rdd.reduceByKey(lambda a, b: a + b)
```

Collect the result and print

```
result = status_code_counts.collect()
```

for status_code, count in result:

```
    print(f"Status Code: {status_code}, Count: {count}")
```

```
>>> log_rdd = spark.sparkContext.textFile("logs_dataset/server-access-log.txt")
>>> for line in contents:
...     print(line)
...
13.66.139.0 - - [19/Dec/2020:13:57:26 +0100] "GET /index.php?option=com_phocagallery&view=category&id=1:almhuette-raith&Itemid=53 HTTP/1.1" 200 32653 "-" Mozilla/5.0 (compatible; b
bot/2.0; +http://www.bing.com/bingbot.htm) "-"
157.48.153.185 - - [19/Dec/2020:14:08:06 +0100] "GET /apache-log/access.log HTTP/1.1" 200 233 "-" Mozilla/5.0 (Windows NT 6.3; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Ch
e/87.0.4280.88 Safari/537.36 "-"
157.48.153.185 - - [19/Dec/2020:14:08:08 +0100] "GET /favicon.ico HTTP/1.1" 404 217 "http://www.almhuette-raith.at/apache-log/access.log" Mozilla/5.0 (Windows NT 6.3; Win64; x64) Ap
leWebKit/537.36 (KHTML, like Gecko) Chrome/87.0.4280.88 Safari/537.36 "-"
216.244.66.230 - - [19/Dec/2020:14:14:26 +0100] "GET /robots.txt HTTP/1.1" 200 304 "-" Mozilla/5.0 (compatible; DotBot/1.1; http://www.opensiteexplorer.org/dotbot, help@moz.com)" -
54.36.148.92 - - [19/Dec/2020:14:16:44 +0100] "GET /index.php?option=com_phocagallery&view=category&id=2K3Awinterfotos&Itemid=53 HTTP/1.1" 200 30662 "-" Mozilla/5.0 (compatible; Ah
sBot/7.0; +http://ahrefs.com/robot/)" "-"
92.101.35.224 - - [19/Dec/2020:14:29:21 +0100] "GET /administrator/index.php HTTP/1.1" 200 4263 "-" Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322)" "-"
73.166.162.225 - - [19/Dec/2020:14:58:59 +0100] "GET /apache-log/access.log HTTP/1.1" 200 1299 "-" Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) C
ome/87.0.4280.101 Safari/537.36 "-"
73.166.162.225 - - [19/Dec/2020:14:58:59 +0100] "GET /favicon.ico HTTP/1.1" 404 217 "http://www.almhuette-raith.at/apache-log/access.log" Mozilla/5.0 (Windows NT 10.0; Win64; x64) /
leWebKit/537.36 (KHTML, like Gecko) Chrome/87.0.4280.101 Safari/537.36 "-"
54.36.148.108 - - [19/Dec/2020:15:09:30 +0100] "GET /robots.txt HTTP/1.1" 200 304 "-" Mozilla/5.0 (compatible; AhrefsBot/7.0; +http://ahrefs.com/robot/)" "-"
54.36.148.1 - - [19/Dec/2020:15:09:31 +0100] "GET /index.php?option=com_phocagallery&view=category&id=2K3Awinterfotos&Itemid=53 HTTP/1.1" 200 30618 "-" Mozilla/5.0 (compatible; Ahref
Bot/7.0; +http://ahrefs.com/robot/)" "-"
>>>
```

```
>>> status_code_rdd = log_rdd.map(lambda line: (line.split(" ")[5].replace("'", ""), 1))
>>> status_code_counts = status_code_rdd.reduceByKey(lambda a, b: a + b)
>>> result = status_code_counts.collect()
>>> for status_code, count in result:
...     print(f"Status Code: {status_code}, Count: {count}")
...
```

Status Code: "POST", Count: 56995

Status Code: "OPTIONS", Count: 1

Status Code: "GET", Count: 41075

Status Code: "HEAD", Count: 307

- **Arrange the result in descending order and display the result**

```
# Map each status code to a tuple, reduce by key, and sort by count in descending order
status_counts_sorted = status_code_counts.sortBy(lambda x: x[1], ascending=False)
```

```
# Collect and print the results in descending order
```

```
for result in status_counts_sorted.collect():
```

```
    print(f"Status Code: {result[0]}, Count: {result[1]}")
```

```
>>> status_counts_sorted = status_code_counts.sortBy(lambda x: x[1], ascending=False)
>>> for result in status_counts_sorted.collect():
...     print(f"Status Code: {result[0]}, Count: {result[1]}")
...
Status Code: "POST, Count: 56995
Status Code: "GET, Count: 41075
Status Code: "HEAD, Count: 307
Status Code: "OPTIONS, Count: 1
```

- **Identify the top 10 frequent visitors of the website and show the result in the RDD**

```
# Extract IP addresses and count occurrences
```

```
ips_counts_rdd = log_rdd.map(lambda line: (line.split(" ")[0], 1)) \
    .reduceByKey(lambda a, b: a + b) \
    .sortBy(lambda x: x[1], ascending=False)
```

```
# Take the top 10 frequent visitors
```

```
top_visitors = ips_counts_rdd.take(10)
```

```
# Print the top 10 visitors
```

```
for ip, count in top_visitors:
```

```
    print(f"IP Address: {ip}, Visits: {count}")
```

```
>>> ips_counts_rdd = log_rdd.map(lambda line: (line.split(" ")[0], 1)) \
...     .reduceByKey(lambda a, b: a + b) \
...     .sortBy(lambda x: x[1], ascending=False)
>>>
>>> top_visitors = ips_counts_rdd.take(10)
>>> for ip, count in top_visitors:
...     print(f"IP Address: {ip}, Visits: {count}")
...
IP Address: 193.106.31.130, Visits: 43423
IP Address: 173.255.176.5, Visits: 5220
IP Address: 178.44.47.170, Visits: 2824
IP Address: 51.210.183.78, Visits: 2684
IP Address: 45.15.143.155, Visits: 1927
IP Address: 45.144.0.179, Visits: 946
IP Address: 176.222.58.254, Visits: 934
IP Address: 45.132.207.154, Visits: 890
IP Address: 45.153.227.55, Visits: 888
IP Address: 45.138.4.22, Visits: 880
```

- Identify the top 10 missing (does not exist) URLs using these steps:
 - Read the log file as an RDD in PySpark
 - Identify the URLs for which the server is returning the 404-request code and display the data

Filter for 404 responses, extract URLs, and count occurrences

```
missing_urls_rdd = log_rdd \
    .filter(lambda line: " 404 " in line) \
    .map(lambda line: (line.split(" ")[10], 1)) \
    .reduceByKey(lambda a, b: a + b) \
    .sortBy(lambda pair: pair[1], ascending=False)
```

Take the top 10 missing URLs

```
top_missing_urls = missing_urls_rdd.take(10)
```

Print the top 10 missing URLs

```
for url, count in top_missing_urls:
    print(f"URL: {url}, Count: {count}")
```

```
>>> missing_urls_rdd = log_rdd \
...     .filter(lambda line: " 404 " in line) \
...     .map(lambda line: (line.split(" ")[10], 1)) \
...     .reduceByKey(lambda a, b: a + b) \
...     .sortBy(lambda pair: pair[1], ascending=False)
>>> top_missing_urls = missing_urls_rdd.take(10)
>>> for url, count in top_missing_urls:
...     print(f"URL: {url}, Count: {count}")
...
URL: "-", Count: 3070
URL: "http://www.almhuetten-raith.at", Count: 609
URL: "http://www.almhuetten-raith.at/", Count: 447
URL: "http://www.almhuetten-raith.at/apache-log/access.log", Count: 398
URL: "http://www.almhuetten-raith.at/apache-log/", Count: 183
URL: "http://almhuetten-raith.at/", Count: 153
URL: "http://www.almhuetten-raith.at/index.php?option=com_phocagallery&view=category&id=1&Itemid=53", Count: 90
URL: "http://www.almhuetten-raith.at/index.php?option=com_content&view=article&id=49&Itemid=55", Count: 68
URL: "http://www.almhuetten-raith.at/index.php?option=com_content&view=article&id=50&Itemid=56", Count: 53
URL: "http://www.almhuetten-raith.at/robots.txt", Count: 51
```

Identify the traffic (total number of HTTP requests received per day) using the below steps:

- Read the log file as an RDD in PySpark
- Fetch the DateTime string and replace "[" with blank
- Get the date string from the DateTime
- Identify HTTP requests using the map function
- Display the data

Extract date and map each date to a tuple (date, 1), then reduce by key to count

```
daily_traffic = log_rdd.filter(lambda line: "HTTP" in line) \
    .map(lambda line: line.split(" ")[3].lstrip("[").split(":")[0]) \
    .map(lambda date: (date, 1)) \
    .reduceByKey(lambda a, b: a + b) \
    .sortByKey()
```

Collect and print the results

```
for date, count in daily_traffic.collect():
    print(f"Date: {date}, Requests: {count}")
```

```
>>> daily_traffic = log_rdd.filter(lambda line: "HTTP" in line) \
...     .map(lambda line: line.split(" ")[3].lstrip("[").split(":")[0]) \
...     .map(lambda date: (date, 1)) \
...     .reduceByKey(lambda a, b: a + b) \
...     .sortByKey()
>>> for date, count in daily_traffic.collect():
...     print(f"Date: {date}, Requests: {count}")
...
Date: 01/Jan/2021, Requests: 2165
Date: 02/Jan/2021, Requests: 1942
Date: 03/Jan/2021, Requests: 2379
Date: 04/Jan/2021, Requests: 2788
Date: 05/Jan/2021, Requests: 2017
Date: 06/Jan/2021, Requests: 2386
Date: 07/Jan/2021, Requests: 3098
Date: 08/Jan/2021, Requests: 4056
Date: 09/Jan/2021, Requests: 2805
Date: 10/Jan/2021, Requests: 2313
Date: 11/Jan/2021, Requests: 4283
Date: 12/Jan/2021, Requests: 2300
Date: 13/Jan/2021, Requests: 2475
Date: 14/Jan/2021, Requests: 1954
Date: 15/Jan/2021, Requests: 2227
Date: 16/Jan/2021, Requests: 2328
Date: 17/Jan/2021, Requests: 2498
Date: 18/Jan/2021, Requests: 4988
Date: 19/Dec/2020, Requests: 1135
Date: 19/Jan/2021, Requests: 2302
Date: 20/Dec/2020, Requests: 3698
Date: 20/Jan/2021, Requests: 2204
Date: 21/Dec/2020, Requests: 3982
Date: 22/Dec/2020, Requests: 3645
```

```
Date: 23/Dec/2020, Requests: 3856
Date: 24/Dec/2020, Requests: 3607
Date: 25/Dec/2020, Requests: 5644
Date: 26/Dec/2020, Requests: 2269
Date: 27/Dec/2020, Requests: 2181
Date: 28/Dec/2020, Requests: 7478
Date: 29/Dec/2020, Requests: 2919
Date: 30/Dec/2020, Requests: 2389
Date: 31/Dec/2020, Requests: 2067
```

- Identify the top 10 endpoints that transfer maximum content in megabytes and display the data

```
# Function to parse log line, extract endpoint and bytes transferred, and convert bytes to megabytes
def parse_log_line(line):
    parts = line.split(" ")
    endpoint = parts[6]
    bytes_transferred = int(parts[-1]) if parts[-1].isdigit() else 0
    megabytes_transferred = bytes_transferred / (1024 * 1024)
    return (endpoint, megabytes_transferred)

# Extract endpoints and bytes transferred, aggregate by endpoint, and sort by total megabytes transferred
top_endpoints_rdd = log_rdd.map(parse_log_line) \
    .reduceByKey(lambda a, b: a + b) \
    .sortBy(lambda pair: pair[1], ascending=False)

# Take the top 10 endpoints
top_endpoints = top_endpoints_rdd.take(10)

# Print the top 10 endpoints and their total megabytes transferred
for endpoint, mb_transferred in top_endpoints:
    print(f"Endpoint: {endpoint}, Total MB Transferred: {mb_transferred:.2f}")
```

```
>>> def parse_log_line(line):
...     parts = line.split(" ")
...     endpoint = parts[6]
...     bytes_transferred = int(parts[-1]) if parts[-1].isdigit() else 0
...     megabytes_transferred = bytes_transferred / (1024 * 1024)
...     return (endpoint, megabytes_transferred)
...
>>> top_endpoints_rdd = log_rdd.map(parse_log_line) \
...     .reduceByKey(lambda a, b: a + b) \
...     .sortBy(lambda pair: pair[1], ascending=False)
24/02/12 12:10:04 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
24/02/12 12:10:04 WARN shortcircuit.DomainSocketFactory: The short-circuit local reads feature cannot be used because libhadoop cannot be loaded.
>>> top_endpoints = top_endpoints_rdd.take(10)
>>> for endpoint, mb_transferred in top_endpoints:
...     print(f"Endpoint: {endpoint}, Total MB Transferred: {mb_transferred:.2f}")
...
Endpoint: /index.php?option=com_phocagallery&view=category&id=1:almhuetite-raith&Itemid=53, Total MB Transferred: 0.00
Endpoint: /apache-log/access.log, Total MB Transferred: 0.00
Endpoint: /robots.txt, Total MB Transferred: 0.00
Endpoint: /index.php?option=com_phocagallery&view=category&id=2%3Aawinterfotos&Itemid=53, Total MB Transferred: 0.00
Endpoint: /administrator/index.php, Total MB Transferred: 0.00
Endpoint: /administrator/%22, Total MB Transferred: 0.00
Endpoint: /templates/system/css/general.css, Total MB Transferred: 0.00
Endpoint: /templates/jp_hotel/css/template.css, Total MB Transferred: 0.00
Endpoint: /templates/jp_hotel/css/layout.css, Total MB Transferred: 0.00
Endpoint: /media/system/js/caption.js, Total MB Transferred: 0.00
>>>
```