# CprE 419 Lab 6: Log File Analysis Using Pig

**Department of Electrical and Computer Engineering**
**Iowa State University**
**Spring 2014**

## Purpose

The goal is to use the Pig platform and the scripting language Pig Latin, to analyze large Log Files and trace files. While you can write a MapReduce program for each such task, Pig can help you do this task faster, since it helps you program at a higher level of abstraction than MapReduce. Scripts written in Pig Latin are automatically converted to MapReduce jobs.

During this lab, you will learn:

- The Pig platform and the scripting language Pig Latin
- Some real world problems that can be solved using Pig

## Submission

Create a single zip archive with the following and hand it in through blackboard**:**

- Each individual output file generated by your program (**except** firewall – it's too big)
- Commented Code for your program. Include all source files needed for compilation.

## Pig

Before you use Pig, you need to set up environment first. Open ~/.bashrc, and add the following two lines in that:

```
PATH="/hadoop/bin:/work/419x/pig/bin:$PATH"
export JAVA_HOME=/usr/lib/jvm/java-1.7.0-openjdk-1.7.0.51.x86_64
```

And then save.

The pig prompt can be obtained by typing 'pig' from **n0 (not cystorm)** and quit by typing 'quit'.

```
$ pig

grunt>
```

To use pig in local mode do:

```
$ pig -x local
```

In the local mode, pig will not use Hadoop to distribute the work across the clusters. Instead it will perform all work directly on the namenode.

But first, let us see an example of a Pig script in fully distributed mode. We will create a Pig script for the Word Count problem and see how it works. Copy the following code into a file called wordcount.pig, replace <NETID> with your net id, and save it in your home directory on cystorm.

**wordcount.pig**

```
input_lines = LOAD '/class/s15419x/lab2/shakespeare' USING
PigStorage('\n') AS (line:chararray);

-- Extract words from each line and put them into a pig bag
-- datatype, then flatten the bag to get one word on each row
words = FOREACH input_lines GENERATE FLATTEN(TOKENIZE(line)) AS
word;

-- filter out any words that are just white spaces
filtered_words = FILTER words BY word MATCHES '\\w+';

-- create a group for each word
word_groups = GROUP filtered_words BY word;

-- count the entries in each group
word_count = FOREACH word_groups GENERATE COUNT(filtered_words) AS
count, group AS word;

-- order the records by count
ordered_word_count = ORDER word_count BY count DESC;
STORE ordered_word_count INTO '/scr/<NETID>/lab6/exp1/output/';
```

Execute the script with the following command from **n0 (not cystorm):**

```
$ pig wordcount.pig
```

The output can be checked via HDFS as for a standard Hadoop job.

Look here for documentation on pig:
http://pig.apache.org/docs/r0.14.0/

The Pig Latin language can be learnt from the link:
http://pig.apache.org/docs/r0.14.0/basic

Pig utilities:
http://pig.apache.org/docs/r0.14.0/cmds

## Experiment 1 (20 points)

For this experiment we will use Pig in local mode and analyze US demographic data.
It can be found at: http://www.census.gov/geo/maps-data/data/gazetteer2010.html

You may download `gaz_tracts_national.txt` from Piazza, or access the file from namenode at `/class/s14419x/lab6/gaz_tracts_national.txt` The file has the following columns:

| Column | Label | Description |
|---|---|---|
| Column 1 | USPS | United States Postal Service State Abbreviation |
| Column 2 | GEOID | Geographic Identifier - fully concatenated geographic code (State FIPS, County FIPS, census tract number) |
| Column 3 | POP10 | 2010 Census population count. |
| Column 4 | HU10 | 2010 Census housing unit count. |
| Column 5 | ALAND | Land Area (square meters) - Created for statistical purposes only. |
| Column 6 | AWATER | Water Area (square meters) - Created for statistical purposes only. |
| Column 7 | ALAND_SQMI | Land Area (square miles) - Created for statistical purposes only. |
| Column 8 | AWATER_SQMI | Water Area (square miles) - Created for statistical purposes only. |
| Column 9 | INTPTLAT | Latitude (decimal degrees) First character is blank or "-" denoting North or South latitude respectively. |
| Column 10 | INTPTLONG | Longitude (decimal degrees) First character is blank or "-" denoting East or West longitude respectively. |

We are interested in the field 'ALAND – Land Area (square meters)'

Write a Pig script and run it in local mode on this data to find out the top 2 states according to the land area. Since you want to execute the script in local mode, do not save your answer in HDFS but in local file system (your cystorm home directory). Include the source code and the output file in the submission.

## Experiment 2 (20 points)

Next we will use Pig in distributed mode. One nice thing about Pig is that the same code can be used in either local or distributed mode. You can run the previous experiment in distributed mode to check the results. You will require no change in the code except for the names of input and output files.

For this experiment we have a network trace file from a network monitor.

It is in the server location: `/class/s15419x/lab6/network_trace` along with a smaller file you can use for testing `/class/s15419x/lab6/test_data`

The file is a real life network trace that was dumped from a network monitor. These files can be queried for information about network traffic.

An example entry in the file is:

```
10:20:00.000020 IP 244.131.189.196.22379 > 245.184.172.199.80: tcp 0
```

The data format is as follows:

<Time> "IP" <Source IP> ">" <Destination IP> <protocol> <protocol dependent data>

The protocol dependent data will be different for TCP, UDP etc. and can be ignored for this lab.

Usually, IP addresses are of the format A.B.C.D However, this data presents IP addresses in the format has the format of A.B.C.D.E. You will have to process the IP to get rid of the extra information including and after the fourth "."

In network monitoring, it is useful to know the identity of those IP sources that connect to a large number of distinct IP destinations. Such sources are often malicious nodes, or compromised by malicious software, and maybe sending spam.

Write a pig script to find the top 10 source IP addresses ranked according to the number of unique destination IP addresses that they connect to using the TCP protocol. Please note that we are not interested in communication that does not use the TCP protocol. The script should be able to automatically save the output in the location:

`/scr/<NETID>/lab6/exp2/output/`

## Experiment 3 (20 points)

Suppose that there was a firewall that blocked IP addresses that it believed were potentially unsafe. The list of all IP connections that were blocked is stored in memory, and also in a log file, but the firewall log was lost, due to a failure. We want to regenerate this logfile from the other data sources that we have. It is important to regenerate this information, as the IP addresses that are blocked regularly are added to a black list.

The lost firewall log contained details of all blocked connections and looks as follows:

<Time> <Connection ID> <Source IP> <Destination IP> "Blocked"

Your task is to regenerate the log file in the above format by combining information from others logs that are available. In particular, we have the following files:

`/class/s15419x/lab6/ip_trace` – An IP trace file having information about connections received from different source IP addresses, along with a connection ID and time.

`/class/s15419x/lab6/raw_block` - A file containing the connection IDs that were blocked

The IP trace file has the following format:

```
0:0:0:9 8 215.160.81.159 > 174.83.200.101 UDP 943
```

The format is similar (but not exactly the same) to the previous experiment

<Time> <Connection ID> <Source IP> ">" <Destination IP> <protocol> <protocol dependent data>

The firewall file has lines in the following format: <Connection ID> <Action Taken>
For instance, it could look as follows:

```
0 Allowed
1 Blocked
2 Allowed
3 Blocked
```

Your task is as follows.

A. Write a Pig script to regenerate the firewall log file. The output file should be at the location `/scr/<NETID>/lab6/exp3/firewall`

B. Use the previous script to generate the list of all unique source IP addresses that were blocked and the number of times that they were blocked. This list should be sorted (by the script) by the number of times each IP was blocked in descending order. The result should be in the location `/scr/<NETID>/lab6/exp3/output`

Finally, your solution should have one script that does both tasks A and B.

For this experiment, it is useful to use the "JOIN" operation. Joining data is a key feature of PIG, and works as follows. Consider the two data sets

Data A: u, x, y, z
Data B: a, b, c, u

These two data sets can be joined using the variable "u" to form: u, x, y, z, a, b, c if A::u == B::u