NetFlow Analysis with MapReduce

Wonchul Kang, Yeonhee Lee, Youngseok Lee
Chungnam National University
{teshi85, yhlee06, lee}@cnu.ac.kr
2010.04.24(Sat)

based on "An Internet Traffic Analysis Method with MapReduce", Cloudman workshop, April 2010
Introduction

• Flow-based traffic monitoring
  – Volume of processed data is reduced

• Traditional flow-based traffic monitoring
  – Run on a high performance central server
Motivation

• A huge amount of flow data
  – Long-term collection of flow data

<table>
<thead>
<tr>
<th># of Routers</th>
<th>1 Day</th>
<th>1 Month</th>
<th>1 Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.2 GB</td>
<td>13 GB</td>
<td>156 GB</td>
</tr>
<tr>
<td>5</td>
<td>6 GB</td>
<td>65 GB</td>
<td>780 GB</td>
</tr>
<tr>
<td>10</td>
<td>12 GB</td>
<td>130 GB</td>
<td>1.5 TB</td>
</tr>
<tr>
<td>200</td>
<td>240 GB</td>
<td>2.6 TB</td>
<td>30 TB</td>
</tr>
</tbody>
</table>

  – Short-term period of flow data
    • Massive flow data from anomaly traffic data of Internet worm and DDoS

• Cluster file system and cloud computing platform
  – Google’s programming model, MapReduce, big table [8]
  – Open-source system, Hadoop [9]
MapReduce

• MapReduce is a programming model for large data set

• First suggested by Google

• User only specify a map and a reduce function
  – Automatically parallelized and executed on a large cluster
MapReduce

- **Map**: return a list containing zero or more \((k, v)\) pair
  - Output can be a different key from the input
  - Output can have same key

- **Reduce**: return a new list of reduced output from input
Hadoop

- Open-source framework for running applications on large clusters built of commodity hardware

- Implementation of MapReduce and HDFS
  - MapReduce: computational paradigm
  - HDFS: distributed file system

- Node failures are automatically handled by framework

- Hadoop
  - Amazon: EC2, S3 service
  - Facebook: analyze the web log data
Related Work

• Widely used tools for flow statistics
  – Flow-tools, flowscan or CoralReef[5]

• P2P-based distributed analysis of flow data
  – DIPStorage: each storage tank associated with a rule [11]

• MapReduce software
  – Snort log analysis: NCHC cloud computing research group [16]
Contribution

• A flow analysis method with MapReduce
  – Process flow data in a cloud computing platform, hadoop

• Implementation of flow analysis programs with Hadoop
  – Decrease flow computation time
  – Enhance fault-tolerant of flow analysis jobs
Architecture of Flow Measurement and Analysis System

- Each router exports flow data to cluster node
- Cluster master manages cluster nodes
Components of Cluster Node

- Flow file input processor
- Flow analysis map/reduce
- Flow-tools
- Hadoop
  - HDFS
  - MapReduce
- Java VM
- OS: Linux

- Cluster File System (HDFS)
- Flow Analysis Map
- Flow Analysis Reduce
- MapReduce Library
- Hadoop
- Java Virtual Machine
- Operating System (Linux)
- Hardware (CPU, HDD, Memory, NIC)
Flow File Input Processor

- Save NetFlow data in binary flow file
- Convert binary flow file into text file
- Copy text file to HDFS
Flow Analysis Map/Reduce

- Read text flow files
- Run map tasks
  - Read each line
  - (Validation Check)
  - Parsing flow data
  - Save result into temporary files (key, value)
- Run reduce tasks
  - Read temporary files (Key, List[Value])
  - Run sum process
- Write results to a file
Performance Evaluation

Environment

- Data: flow data from /24 subnet

<table>
<thead>
<tr>
<th>Duration</th>
<th>Flow count (million)</th>
<th>Flow file count</th>
<th>Total binary file size (GB)</th>
<th>Total text file size (GB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 day</td>
<td>3.2</td>
<td>228</td>
<td>0.2</td>
<td>1.2</td>
</tr>
<tr>
<td>1 week</td>
<td>19.0</td>
<td>1596</td>
<td>0.3</td>
<td>2.3</td>
</tr>
<tr>
<td>1 month</td>
<td>109.1</td>
<td>7068</td>
<td>2.0</td>
<td>13.1</td>
</tr>
</tbody>
</table>

- Compared methods: computing byte count per destination port
  - Our implementation with Hadoop

- Performance metric
  - flow statistics computation time

- Fault recovery against map/reduce tasks
Our Testbed

- Hadoop 0.18.3
- Cluster master x 1
  - Core 2 Duo 2.33 GHz
  - Memory 2GB
  - 1 GE
- Cluster node x 4
  - Core 2 Quad 2.83 GHz
  - Memory 4GB
  - HDD 1.5 TB
  - 1 GE
Flow Statistics Computation Time

- Port breakdown computation time
  - 72% decrease with MR(4) on Hadoop
Single Node Failure : Map Task

- Under 4 cluster nodes
- Map task fail time
  - 4 sec (M : 9% R : 0%)
- Map task recover time
  - 266 sec (M : 99% R : 32%)
Single Node Failure : Reduce Task

- Under 4 cluster nodes
- Reduce task fail time
  - 29 sec (M : 41% R : 10% )
- Reduce task recover time
  - 320 sec (M : 99% R : 32% )
Text vs. Binary NetFlow Files

Flow analysis with text files

Flow analysis with binary files
Binary Input in Hadoop

- Currently developing **BinaryInputFormat** module for Hadoop

- Small storage by binary NetFlow files
  - Reduces # of Map tasks → increasing performance

- Decreasing computation time
  - By 18% ~ 55% for a single flow analysis job
  - By 58% ~ 75% for two flow analysis jobs
Prototype
Summary

• NetFlow data analysis with MapReduce
  – Easy management of big flow data
  – Decreasing computation time
  – Fault-tolerant service against a single machine failure

• Ongoing work
  – Supporting binary NetFlow files
  – Enhancing fast processing of NetFlow files
References