Problem Solving Hands-on Labware for Teaching Big Data Cybersecurity Analysis

Teng Zhao, Kai Qian, Dan Lo, Minzhe Guo, Prabir Bhattacharya, Wei Chen, and Ying Qian

***Abstract*—The rapid growth of big data analysis needs has resulted in a shortage of professionals in this area. Despite such great need, big data analytics and data science is not well represented in current academic programs. Although a few schools have offered new courses in this area but they all face challenges. Most of existing instruction models are impractical only focus on lectures, seminars, and discussions without any hands-on problem solving computing labs because it requires significant investment in resources and high requirement for instructors (e.g., faculty whose expertise is in this area). To overcome the above difficulties this paper presents a new labware with real hands-on labs to support the course objectives. The designed labware is inexpensive, portable, and easy-to-adopt.**

***Index Terms*— Big data, analysis, security, lab experimentation**

I. Introduction

The recent explosion of cyber activities in the areas of social network, healthcare, research, and many other business resulted in huge unstructured data(big data), most of them are unstructured in various forms. These data may provide very valuable information, such as for bioinformatics research, security and business analysis, for exploring new knowledge, patterns, and relevant information. The rapid growth of big data analysis has resulted in a shortage of professionals in this area. Despite such great need, big data analytics and data science is not well represented in our current academic computing programs. Although a few schools have started offering new courses in this area but they face challenges. It is very crucial for students in data science and computing area to know how to manipulate, store, and analyze big data with computing technology.

Most of existing instruction models focus on lectures, seminars, discussions without any hands-on computing labs because it rely on two prerequisites, i.e., significant investment in resources (cloud services) and high requirement for instructors (e.g., faculty whose expertise is in this area).

T. Zhao, K. Qian, and D. Lo are with the Dept. of Computer Science and Software Engineering, University of Southern Polytechnic State University, Marietta, GA 30060 USA (e-mail: tzhao@spsu.edu, kqian@spsu.edu, and dlo@spsu.edu).

M. Guo and P. Bhattacharya are with Dept. of Computer Science, University of Cincinnati, Cincinnati, OH 45220 USA (email: guome@mail.uc.edu, and bhattapr@ucmail.uc.edu).

W. Chen is with Dept. of Computer Science, Tennessee State University, Nashville, TN 37209 USA (e-mail: wchen@tnstate.edu).

Y. Qian is with Dept. of Computer Science & Technology, East China Normal University, Shanghai, China (e-mail: yqian@cs.ecnu.edu.cn)

There is a need to design an adoptable open source laboratory model to be used in big data analysis teaching and learning to overcome the above difficulties to achieve broader big data analysis education, this project aims to develop a new hands-on labware for big data processing and analysis that is portable, modular, and easy-to-adopt. We have adopted an open source framework Hadoop with NoSQL for data store to store big volume unstructured dataset, and MapReduce techniques for parallel processing and for our labware.

II. Big Data Analysis for Information Security Course

A. Course Objectives and Learning Outcomes

OBJECTIVES:

- To explore the fundamental concepts of big data analytics
- To learn the big data analysis with intelligent reasoning and data mining
- To understand the applications using Map Reduce Concepts.
- To apply the big data analysis to cybersecurity

LEARNING OUTCOMES (LOs):

The students will be able to:

- LO1: Work with big data platform
- LO2: Work with NoSQL database for storing unstructured big data
- LO3: Use the HADOOP and Map Reduce technologies associated for big data analytics
- LO4: Design algorithms for big data mining for business applications especially cybersecurity threat analysis

B. Course Description

A project-oriented hands-on course focusing on big data application development for information security over distributed environments. Students will explore key data analysis and management techniques applied to massive network traffic datasets to support real-time decision making for security threats in distributed environments. This course is designed to be a hands-on learning experience that students learn better by doing. With concrete, practical experience students will be better prepared to apply their new knowledge into real-life, data-intensive, research situations. Students are expected to make of map-reduce parallel computing paradigm and associated technologies such as distributed file systems, Mongo databases, and stream computing engines to design scalable big data analysis system. Students should be able to apply machine

ISBN: 978-988-19252-0-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

leaning, supervised or unsupervised-learning, information extraction and feature selection and stream mining in information security domains.

III. Hands-on Laboratory

To achieve the course objectives and student learning outcomes we designed and implemented the hands-on distributed and parallel big data analysis labware.

A. *Lab-1: Apache Hadoop Configuration*

Lab-1 mainly introduce a big data platform, Apache Hadoop, to students and allow them to understand how Hadoop and a MapReduce task works. This lab also provides students with step-by-step instructions, which allows them to be able to configure and work with it, which satisfies the LO1 learning outcome.

The Apache Hadoop software is an open-source framework built for reliable, scalable distributed computing tasks with huge data sets over a cluster of multiple computers. This framework is written in Java programming language and it is under Apache License. Generally, a Hadoop system is composed of a computer acting as the master node and multiple computers acting as the slave nodes, as shown in Fig 1.

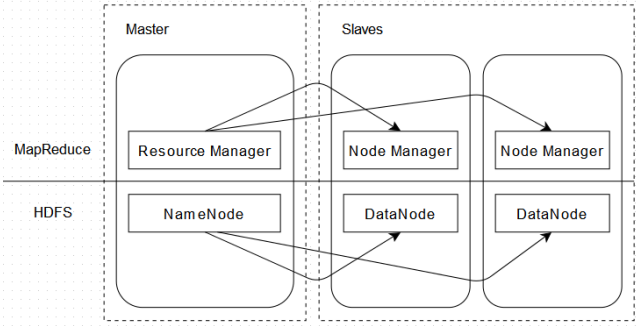

Fig 1. Architecture of Hadoop 2.3.0 Cluster

Hadoop has two modules in total, including HDFS (Hadoop Distributed File System) and MapReduce Framework. HDFS usually only has one NameNode, which is used to manage the directory tree and the metadata of related files of HDFS. It could also own a Secondary NameNode, which can be employed to backup mirror files and to combine logs and mirror files periodically and send back to NameNode. In general, NameNode and Secondary NameNode are deployed on the master node. In addition, DataNode of HDFS is responsible for store data and sending processed data back to NameNode, and it is usually deployed on the slave node.

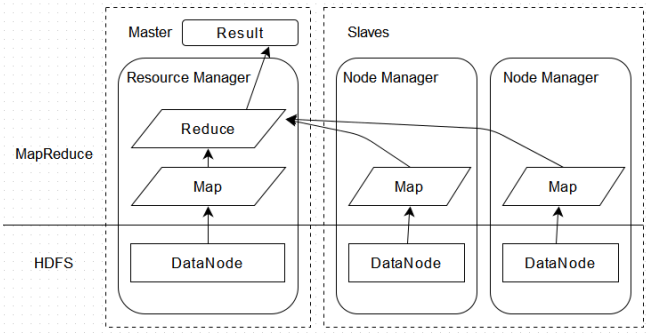

Fig 2. MapReduce Process in Hadoop

After version 0.23, Hadoop starts to adopt new MapReduce framework, called Yarn, consisting of Resource Manager and Node Manager. Resource Manager takes responsibility for managing and dispatching resources in the Hadoop cluster, and receiving data from Node Manager of each slave node. Node Manager is used to send information about usage of resources and task progress to Resource Manager. As shown in Fig 2, MapReduce task is executed in a distributed manner. Each DataNode passes original data to a map function. After being processed by map function, original data becomes key-value pairs and is sent back to a reduce function. After completing the step of reduce function, the data will be saved as result or for further processing.

In this lab, the Hadoop cluster is consisted of three laptops, which help students to get a sense of distribute and parallel computing. One laptop acts as the master node, and the other two laptops are used as slave nodes. After following the tutorial to configure the Hadoop cluster, students can use the following command to start it, as shown in Fig 3.:

```
hadoop@masternode:~/hadoop-binaries/hadoop-2.3.0/sbin$ ./start-all.sh
```

Fig 3. Command For Starting Hadoop Cluster

Then, students can use the following command to check if the Hadoop Cluster is running successfully, as shown in Fig 4:

```
hadoop@masternode:~/hadoop-binaries/hadoop-2.3.0/sbin$ jps
```

Fig 4. Command For Checking Hadoop Cluster

Fig 5, Fig 6, and Fig 7 display the running condition of each laptop. For master node laptop, it should display "Jps, " "NameNode, " "Secondary NameNode, " and "Resource Manager." And for slave node laptops, they should display "Jps, " "DataNode, " and "Node Manager."

```
hadoop@masternode:~/hadoop-binaries/hadoop-2.3.0/sbin$ jps
3232 NameNode
3448 SecondaryNameNode
3605 ResourceManager
3857 Jps
```

Fig 5. Master Node Running Condition

```
hadoop@testnode:~$ jps
3180 Jps
2602 DataNode
2705 NodeManager
```

Fig 6. Slave-1 Node Running Condition

```
hadoop@testnode2:~$ jps
3041 NodeManager
2907 DataNode
3187 Jps
```

Fig 7. Slave-2 Node Running Condition

B. *Lab-2: NoSQL Database - Apache HBase*

Lab-2 concentrates on the introduction of a NoSQL database, Apache HBase. This lab also offers detailed configuration tutorial to students, which can help them to set up HBase on their laptop. In addition, students will be provided with simple instructions of how to use HBase Shell to create table, insert rows, display contents, etc. This lab perfectly meets the requirement of LO2 learning outcome.

ISBN: 978-988-19252-0-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

The Apache HBase is an open-source NoSQL, highly reliable, highly efficient, row-oriented and expandable distributed database system. It is implemented based on BigTable, an open-source software of Google. Similar to the fact that Google BigTable utilizes GFS as its file storage system, HBase employs Hadoop HDFS as its own file storage system; Google executes MapReduce to process huge data sets in BigTable, and HBase runs Hadoop MapReduce; Google BigTable takes advantage of Chubby as cooperative service, and HBase utilizes Zookeeper as its own cooperative service.

In this lab, HBase is chosen to introduce to the students because as a NoSQL database, HBase can be easily used to store huge unstructured data, and as Hadoop HDFS provides a reliable low-level storage support for HBase, it is great for processing huge data sets using MapReduce in later labs. After following the tutorial to configure HBase, students can use the following commands to start Hbase service as shown in Fig 8:

```
hadoop@masternode:~$ start-hbase.sh
starting master, logging to /usr/local/hbase/logs/hbase-hadoop-master-masternode
.out
```

Fig 8. Command For Starting HBase

Then, students can start HBase Shell using the following commands, as shown in Fig 9:

```
hadoop@masternode:~$ hbase shell
2014-06-23 12:26:27,643 INFO  [main] Configuration.deprecation: hadoop.native.li
b is deprecated. Instead, use io.native.lib.available
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 0.98.3-hadoop2, rd5e65a9144e315bb0a964e7730871af32f5018d5, Sat May 31 19
:56:09 PDT 2014

hbase(main):001:0>
```

Fig 9. Command For Starting HBase Shell

After starting HBase Shell, students can use the following commands to perform database manipulation. As shown in Fig 10, students can use 'create' to create table and 'list' to make sure the table is created.

```
hbase(main):011:0> create 'TEST_TABLE', 'Col_1', 'Col_2'
0 row(s) in 0.3440 seconds

=> Hbase::Table - TEST_TABLE
hbase(main):012:0> list
TABLE
TEST_TABLE
1 row(s) in 0.0140 seconds

=> ["TEST_TABLE"]
```

Fig 10. Command For Creating Table in HBase Shell

As shown in Fig 11, students can use 'put' to insert one row with a 'TEST_VALUE' in column 'Col_1.'

```
hbase(main):013:0> put 'TEST_TABLE', 'TEST_ROW', 'Col_1', 'TEST_VALUE'
0 row(s) in 0.0880 seconds
```

Fig 11. Command For Insert Rows in HBase Shell

As shown in Fig 12, students can use 'get' to see the row that they just inserted to the table.

```
hbase(main):014:0> get 'TEST_TABLE', 'TEST_ROW'
COLUMN                CELL
 Col_1:               timestamp=1403541646919, value=TEST_VALUE
1 row(s) in 0.0200 seconds
```

Fig 12. Command For Display Contents in HBase Shell

C. *Lab-3: Word Count Application*

Lab-3 is designed for students to test the cloud cluster they have just configured, and get a sense of how a MapReduce task is executed. In this lab, Word Count application is selected because it is a traditional MapReduce example, which can be easily absorbed by students. This lab will help students to gain LO3 learning outcome.

This lab is developed in Eclipse IDE, and it only contains one source file:

- **WordCountLab.java.**
  This file will be composed of three parts, including Mapper, Reducer and main function.

The MapReduce algorithm of this lab is displayed below in Fig 13:

Algorithm WordCountLab

Map (line_number, words)
  for each word in words
    context.write(word, 1)

Reduce (word, counts[])
  total = 0
  for each count in counts
    total = total + count
  context.write(word, total)

Fig 13. MapReduce Algorithm for WordCountLab

After implementing this source file in Eclipse IDE, students should follow previous instructions to firstly start the Hadoop cluster and configure the Hadoop plugin in Eclipse IDE so that the MapReduce task can be executed directly by Eclipse IDE. The result of Word Count application is displayed in Fig 14(a), (b), (c):

```
hbase(main):001:0> scan 'WordCount'
```

Fig 14 (a). Commands For Checking Word Count Result

```
ROW                   COLUMN+CELL
 a                    column=content:count, timestamp=1403541926495, value=1
 another              column=content:count, timestamp=1403541926440, value=2
 are                  column=content:count, timestamp=1403541926440, value=1
 brazil               column=content:count, timestamp=1403541926440, value=2
 chat                 column=content:count, timestamp=1403541926365, value=2
 cup                  column=content:count, timestamp=1403541926495, value=1
 day                  column=content:count, timestamp=1403541926440, value=2
 doing                column=content:count, timestamp=1403541926495, value=1
```

Fig 14 (b). Word Count Application Result in HBase - Part 1

```
 today                column=content:count, timestamp=1403541926365, value=1
 too                  column=content:count, timestamp=1403541926365, value=1
 video                column=content:count, timestamp=1403541926495, value=1
 wang                 column=content:count, timestamp=1403541926440, value=2
 world                column=content:count, timestamp=1403541926440, value=2
 you                  column=content:count, timestamp=1403541926495, value=1
 zhao                 column=content:count, timestamp=1403541926365, value=2
28 row(s) in 0.4400 seconds
```

Fig 14 (c). Word Count Application Result in HBase - Part 2

D. *Lab-4: Big Data Analysis for Security*

Lab-4 is using our configured cloud cluster to process a large set of network logs and to analyze if the host machine is attacked with DDoS. This lab is designed for students to get a further better understanding of how to implement a MapReduce task to perform big data security analysis. The method of detecting DDoS attack is Count-Based method[7]. This program will be run over a data set of network logs that has no DDoS attacks, and a threshold will be noted down as the measure of detecting DDoS attack. When this program is used to run over a data set of network logs that has DDoS attacks, it will try to analyze the number of TCP requests, if the number is larger than the threshold, this host will be marked as "In Danger; " If the number is below the threshold, this host will be marked as "Normal." This lab flawlessly helps students to accomplish LO4 learning outcome.

ISBN: 978-988-19252-0-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

Fig 15. shows the whole architecture of this lab. In the very beginning, students should download 2000 DARPA Intrusion Detection Scenario-Specific Data Sets From MIT Lincoln Laboratory. Secondly, students will use WireShark software to convert dump log files to text log files, which will make it easy to abstract log data.

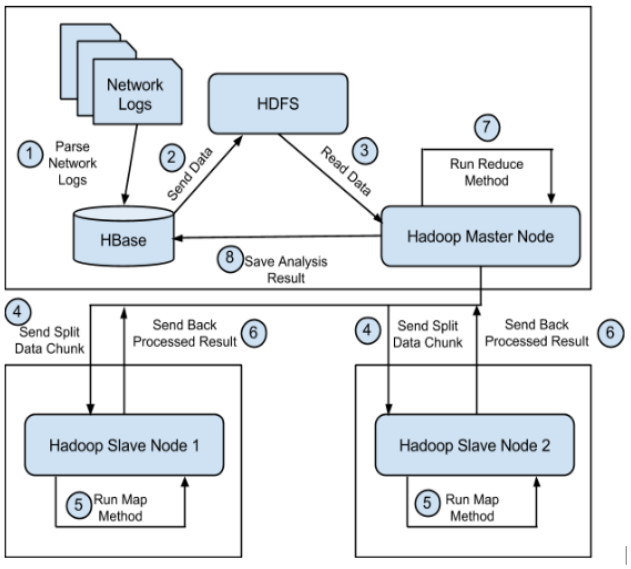

Fig 15. Network Log Security Analysis Workflow

The following steps are listed below:

1. Parse text network log files, and Save data into HBase. (Select timestamp, destination, and request type from each line of log files.)
2. Read Data from HBase to Hadoop HDFS.
3. Hadoop Master Node Read Data From Hadoop HDFS.
4. Split Data Chunk and Send them to Slave Nodes.
5. Execute Map Function on each Slave Node.
6. Send Back Processed Result From Slave Nodes back to Master Node.
7. Execute Reduce Function
8. Analyze Reduce Function Result and Save Result into a result table of HBase

Lab-4 project includes 5 files:

- **Constants.java.**
  This file defines constants and the threshold.
- **HBaseUtil.java.**
  This file defines and implements HBase basic operations, such as create and delete tables, get row from or put row into HBase database.
- **StaticSecurityLogMapReduceAnalysis.java.**
  This file implements the MapReduce algorithm of this lab, act as the starting point of this lab, as shown in Fig 16.
- **StaticLogParser.java**
  This file reads the data from text network log files and save it into HBase NoSQL database.
- **MapReduceResultAnalyzer.java**
  This file analyzes the MapReduce task result and evaluate each host to see if it is attacked by DDoS.

Algorithm BigDataSecurityAnalysisLab

```
Map (time_stamp, hbase_row)
  request_type = hbase_row['type']
  destination_ip = hbase_row['dest']
  if request_type == 'TCP'
      context.write(destination_ip, 1)

Reduce (destination_ip, counts[])
  total = 0
  for each count in counts
    total = total + count
  context.write(destination_ip, total)

Analyze (destination_ip, total)
  if total > threshold
      hbase.put(destination_ip, 'In Danger')
  else
      hbase.put(destination_ip, 'Normal')
```

Fig 16. MapReduce & Analysis Algorithm for BigDataSecurityAnalysisLab

After implementing these five source files in Eclipse IDE, students should follow previous tutorials to firstly start the Hadoop cluster and HBase NoSQL database, and they should be able to execute this application on the cloud cluster of Hadoop and HBase.

In Fig 17, it shows the parsed network log files data in HBase:

```
hbase(main):005:0> scan 'StaticSecurityLog'
```

Fig 17 (a). Commands for Checking Parsed Network Log File Data in HBase

```
 99.999693            column=protocol:, timestamp=1403542125297, value=TCP
 99.999755            column=dest:, timestamp=1403542125297, value=131.84.1.31
 99.999755            column=protocol:, timestamp=1403542125297, value=TCP
 99.999793            column=dest:, timestamp=1403542125297, value=131.84.1.31
 99.999793            column=protocol:, timestamp=1403542125297, value=TCP
 99.999915            column=dest:, timestamp=1403542125297, value=131.84.1.31
 99.999915            column=protocol:, timestamp=1403542125297, value=TCP
74480 row(s) in 27.9900 seconds
```

Fig 17 (b). Parsed Network Log File Data in HBase

In Fig 18, it shows the result of this big data security analysis application in HBase:

```
hbase(main):004:0> scan 'SecurityAnalysisResult'
ROW                  COLUMN+CELL
 131.84.1.31         column=num_of_request:num, timestamp=1403542130156, value=73643
 131.84.1.31         column=status:, timestamp=1403542130358, value=In Danger
 172.16.115.20       column=num_of_request:num, timestamp=1403542130156, value=166
 172.16.115.20       column=status:, timestamp=1403542130358, value=Normal
 172.16.115.5        column=num_of_request:num, timestamp=1403542130156, value=198
 172.16.115.5        column=status:, timestamp=1403542130358, value=Normal
 172.16.116.201      column=num_of_request:num, timestamp=1403542130156, value=89
 172.16.116.201      column=status:, timestamp=1403542130358, value=Normal
 202.77.162.213      column=num_of_request:num, timestamp=1403542130156, value=12
 202.77.162.213      column=status:, timestamp=1403542130358, value=Normal
5 row(s) in 0.0350 seconds
```

Fig 18. Result of Big Data Security Analysis Application

The row name is the destination IP of request, and each row has two columns, one is the number of request, and the other is the evaluation status. When the number of request is over the threshold, the status would become 'In Danger'. Network Administrator could utilize this application analyze DDoS security threats by running it against network log files to see if any host IP was attacked.

IV. Conclusion

The key contribution of our work is to provide hands-on lab environment model and four lab assignments for learning big data analysis from easily configuring Hadoop and HBase environment, to execute example Word Count application, and finally to implement big data security analysis MapReduce application. After taking this course,

ISBN: 978-988-19252-0-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

students will gain the ability to design highly scalable systems that can accept, store, and analyze large volumes of unstructured data.

REFERENCES

[1] Yasin N. Silva, Suzanne W. Dietrich, Jason M. Reed, Integrating Big Data into the Computing Curricula, ACM SIGCSE 2014 http://www.public.asu.edu/~ynsilva/publications/IntegratingBigData.pdf
[2] An Undergraduate Degree in Data Science: Curriculum and a Decade of Implementation Experience, ACM SIGCSE 2014
[3] Philip Sheridan Buffum, CS Principles Goes to Middle School: Learning How to Teach "Big Data", http://people.engr.ncsu.edu/keboyer/papers/buffum-sigcse-2014.pdf, ACM SIGCSE 2014
[4] Nadeem Hamid, Steven Benzel, Towards Engaging Big Data for CS1/2
[5] ACM CSTA, Getting Ready for Big Data, http://blog.acm.org/archives/csta/2013/11/post_27.html
[6] JeongJin Cheon, Tae-Young Choe, Distributed Processing of Snort Alert Log using Hadoop, International Journal of Engineering and Technology 2013.
[7] Yeonhee Lee, Youngseok Lee, Detecting DDoS attacks with Hadoop, CoNEXT '11 Student Proceedings of The ACM CoNEXT Student Workshop

ISBN: 978-988-19252-0-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)