Monitoring and Notification System Based on Risk Analysis Using Map-Reduce Framework

Yoondeuk Seo and Jinho Ahn

***Abstract*—This paper presents a monitoring and notification system based on risk analysis using map-reduce framework that can do risk prediction using big data. By using social media and access records of their users, the proposed system determines their risks. Also, it monitors the restricted area through setting them and notifies the administrator as soon as the intrusions are detected. So, it will provide a control situation for administrator in real time and can detect the risky moments that may occur in advance. Therefore it can reduce the probability of risk of security penetration. Also, since it receives the control situation and provides real-time control screen via smartphone, it is possible to improve the convenience of intrusion management in a very fast and effective way.**

***Index Terms*—Mapreduce, mahout, big data, predictive analytics.**

I. Introduction

Big data and its analysis are at the center of modern science and business. These data are generated from online transactions, emails, videos, audios, images, click streams, logs, posts, search queries, health records, social networking interactions, science data, sensors and mobile phones and their applications [1], [2]. Big data concept means a large dataset which continues to grow so much that it becomes difficult to manage it using existing database management concepts & tools [3]. The difficulty can be related to data capture, storage, search, sharing, analytics and visualization etc. Big data can help to gain insights and make better decisions. It presents an opportunity to create unprecedented business advantage and better service delivery. It also requires new infrastructures and new ways of thinking about how business and IT industry works.

Much of the current discussion about big data analytics today focuses on managing and analyzing unstructured data from business and social sources such as e-mail, videos, tweets, Facebook posts, reviews, and Web behavior [4]-[6]. While this type of big data analytics promises to provide significant value to organizations, data generated at the edge of the network from sensors and other devices represent another huge and untapped resource with the potential to deliver insights that can transform the operations and strategic initiatives of public and private sector organizations [7].

Manuscript received September 8, 2014; revised October 21, 2014. This work was supported in part by the U.S. Department of Commerce under Grant BS123456 (sponsor and financial support acknowledgment goes here).

The authors are with the Department of Computer Science, Kyonggi University, Iuidong, Yeongtonggu, Suwon 443-760 Gyeonggi, Republic of Korea (Corresponding author: Jinho Ahn; tel.: +82 31 249-9674; fax: +82 31 249-9673; e-mail: seoyd@kgu.ac.kr, jhahn@kgu.ac.kr).

This paper presents a monitoring and notification system based on risk analysis using map-reduce framework that can do risk prediction using big data. By using social media and access records of their users, the proposed system determines uncertain risks they could create. By using the risk information that has been obtained from the platform, it can determine whether they should be granted the requested rights to access to restricted areas. And it can set some places that should be monitored as restricted areas. If the intrusion is detected in an area that has been set, then it promptly notifies the administrator with agile viewers showing him/her the intrusion for preparing the next step. So, it will provide a control situation for the administrator in real time and can detect the risky moments that may occur in advance. Therefore it can reduce the probability of risk of security penetration. Also, since it receives the control situation and provides real-time control screen via smartphone, it is possible to improve the convenience of intrusion management in a very fast and effective way.

II. Related Work

The MapReduce programming model is designed to process large volumes of data in parallel by dividing a job into a set of independent tasks [8]-[10]. The job is referred to here as a full MapReduce program, which is the execution of a Mapper or Reducer across a set of data. A task is an execution of a Mapper or Reducer on a slice of data. So, the MapReduce job usually splits the input data set into independent chunks, which are processed by the map tasks in a completely parallel manner.

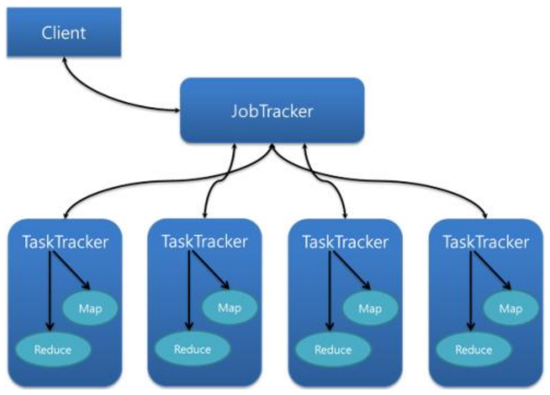

Fig. 1. Hadoop architecture.

Fig. 1 illustrates the Hadoop architecture. The Hadoop MapReduce framework consists of a single Master node that runs a JobTracker instance which accepts Job requests from a client node and Slave nodes each running a TaskTracker instance [11]. The JobTracker assumes the responsibility of distributing the software configuration to the Slave nodes, scheduling the job's component tasks on the TaskTrackers, monitoring them and reassigning tasks to the TaskTrackers when they failed. It is also responsible for providing the status and diagnostic information to the client. The TaskTrackers execute the tasks as directed by the JobTracker. The TaskTracker executes tasks in separate java processes so that several task instances can be performed in parallel at the same time.

The MapReduce input data typically come from the input files loaded into the HDFS. These files are evenly distributed across all the nodes in the cluster. In Hadoop, computer nodes and data nodes are all the same, meaning that the MapReduce and HDFS run on the same set of nodes. At the mapping phase, the input file is divided into independent Input Splits and each split of these Splits describes a unit of work that comprises a single map task in the MapReduce job. The map tasks are then assigned to the nodes in the system based on the physically residence of the input file splits. Several map tasks can be assigned to an individual node, which attempts to perform as many tasks in parallel as it can. When the mapping phase has completed, the intermediate outputs of the map tasks are exchanged between all nodes; and they are also the input of the reduction tasks. This process of exchanging the map intermediate outputs is known as the shuffling. The reduce tasks are spread across the same nodes in the cluster as the mappers. The output of the reduce tasks is stored locally on the slave node.

Technically, a MapReduce system is a framework for processing data in chunks. A framework is a collection of functions that can be called by user code, and that may also call user-defined functions, which are then named callback functions. A MapReduce system has the following properties:

1) The format of the input data can be chosen freely.
2) The output data consist of pairs of arbitrary keys and values.
3) Processing happens in two consecutive phases, using two user-defined functions: mapper and reducer.
4) The mapper function creates intermediate results (pairs of keys and values of arbitrary type each) from each input chunk.
5) The reducer function is applied in unison to all intermediate results with the same key, and produces arbitrarily many final results.

Additionally, there is a main function in a MapReduce framework that has to be called by the user to specify the MapReduce program to be run as a job: which mapper and reducer function to execute and which data to process. MapReduce employs multiple used-defined functions operating on arbitrary data types. For aMapReduce program to be correct, the types of the data items and of the functions need to be compatible with each other.

MapReduce owes its name to the two main phases into which its execution can be divided: the Map phase (in which mainly the mapper function executes) and the Reduce phase (executing the reducer function). This is illustrated in Fig. 2. Rectangles represent chunks of distributed (input, intermediate, and output) data, and ovals labelled "worker" represent nodes executing user-defined functions. In each phase, processing can happen in parallel. In contemporary applications, the computation is typically distributed over a cluster of hundreds to thousands of worker nodes, controlled by a single master node. In this distributed setting, large sets of data have to be serialized (converted to a representation suitable for transport over a network), communicated over the network, and deserialized during a MapReduce computation. As this may incur huge communication costs, MapReduce also contains a feature that optimizes locality in the Map phase: the mapper function, operating on a particular chunk of input data, is typically computed on the same node on which the chunk is stored.

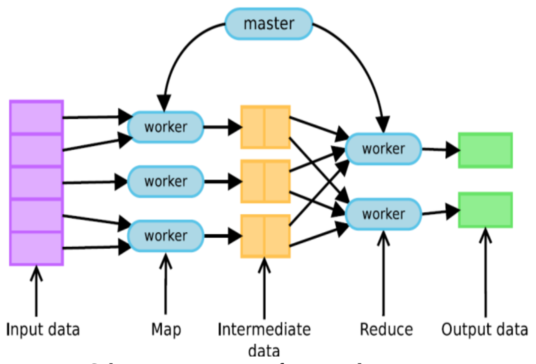

Fig. 2. Schematic overview of MapReduce processing.

Mahout [12] is a machine learning library that runs over a Hadoop system. It has a collection of algorithms to solve clustering, classification and prediction problems. It uses MapReduce paradigm which in combination with Hadoop can be used as an inexpensive solution to solve machine learning problems.

Mahout is a project still in development which has been used mainly for recommendation engines, document classifiers and to solve other web typical problems. At the moment, its usage for clustering is not sufficiently explored.

Hadoop and Mahout are free and open source projects. Due to their inexpensive and scalable characteristics, these platforms can be a promising technology to solve data intensive problems which were not trivial in the past.

However, it is important to study the tradeoff between the overhead of using Map/Reduce and the gain of performance by distributing the computation. It is also essential to check if the clusters maintain their quality.

Mahout contains various implementations of clustering, like K-means, fuzzy K-means, meanshift and Dirichlet among others. To input the data for Mahout clustering it is necessary to do some procedures first. If the data is not numerical it has to be first preprocessed. It is required then to create vectors. If the data set is sparse it allows the user to create sparse vectors that are much more compact. The vectors are finally converted to a specific Hadoop file format that is SequenceFile. The K-means clustering algorithm takes the following input parameters:

- A SequenceFile containing the input vectors.
- A SequenceFile containing the initial cluster centers. If not present it will attribute them randomly.
- A similarity measure to be used.
- The convergence Threshold that will be the stopping condition of the K-means. If in a particular iteration, any

centers of the clusters do not change beyond that threshold, then no further iterations are done.

- The maximum number of iterations to be processed if it does not converge first.
- The number of reducers to be used. This value determines the parallelism of the execution.
- The vector implementation used for the input files.

As output the user gets the centroids coordinates and the samples attributed to each cluster. The output files are in SequenceFile format. Mahout provides the necessary tools for file conversion and creating the vectors.

III. The Proposed Risk Analysis Module

In this section, the risk prediction module is proposed. It was developed by using the Apache mahout. Fig. 3 shows the risk determining process.

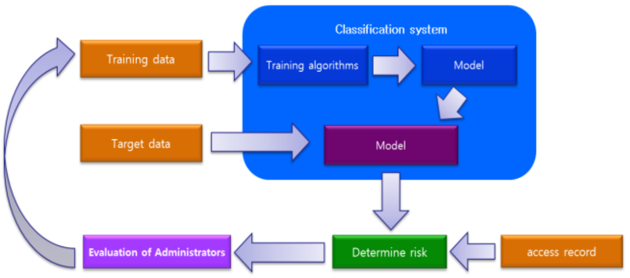

Fig. 3. Risk determining process.

The risk is determined through the following process. First, it creates a training data that contains the predictor value and the target value. Then, it learns the training data through their appropriate training algorithm such as Naive Bayes algorithm. It generates a model through the learning procedure. Using the generated model, it classifies the target data and determines potential risks caused by their users utilizing the access records, the information of social media and the values that have been classified before. The administrator evaluates the classified values and adds them to the training data. Afterwards, it generates more appropriate model using the newly changed training data and executes the same procedural steps as mentioned earlier again.

Training data is input data for creating the model. Target data is obtained by pre-treating the social media of the user. Determine risk is a module for determining potential risks caused by their users utilizing the access records with the results obtained from the classification system.

IV. Platform Architecture

Fig. 4 shows the system architecture. Camera Module controls the function associated with the camera. Image Recognition recognizes the face of the users. It sends the user information which has been recognized to the server. Motion Recognition detects a motion on the screen, sends the camera information to the server. Streaming module provides a streaming service.

Integrated Control Platform manages the control situations and determines the risk using big data. Control data processing manages and processes control messages. Determine Risk determines the risk of the user. MapReduce and HDFS Connectors are responsible for communication with the big data platform. Registration Management stores the control data in the DB. ActiveMQ sends a control message to the Integrated Control Client. Android push sends a control message to the Android mobile phone.

Integrated Control Client is a client program that allows administrators to use the Integrated Control Platform. Registration Management provides a screen for registering the information for the operators. Control Situation Management provides a screen to manage the control situation. ActiveMQ receives the messages from the Integrated Control Platform. Streaming shows the image that has been transferred from the camera.

Big Data platform performs two functions. One is collecting big data. Another is processing big data. In order to determine risk, Data Collector module collects the social media. Data Processing module processes the data from the classification system.

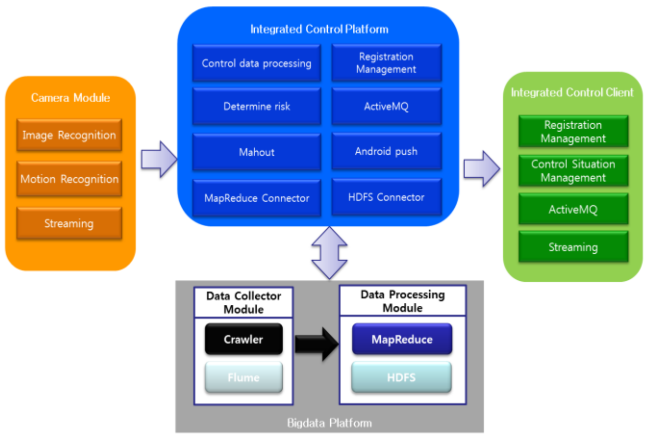

Fig. 4. System architecture.

V. Applied Scenarios of Implemented System

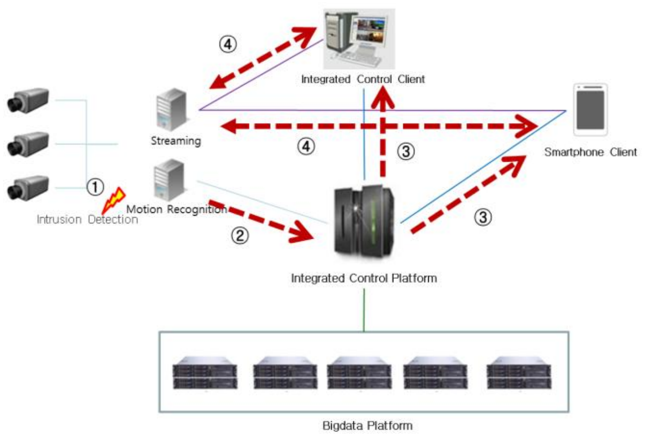

Fig. 5. Intrusion detection.

In this section, the two scenarios are introduced in the proposed system. The first scenario is for intrusion detection like in Fig. 5. By setting the monitored area in the camera image, the monitoring is started. If suspicious motions are detected in the monitored area, then the motion recognition module sends the information of current status to the

Integrated Control Platform. Integrated Control Platform has received the control message and then it transmits the status to the client. When the client receives the message, the client requests the streaming server in order to confirm the current status. The streaming server provides the current video to each requesting client.

The second scenario is for prompt risk notification like in Fig. 6. Big data platform collects Twitter messages periodically. When Collection is completed, then Big data platform reports the situation to the Integrated Control Platform. Integrated Control Platform initiates the risk determination process to have a message that has been collected. If suspicious risks are detected, Integrated Control Platform would send a message related to them to the corresponding client. Upon receiving the message from the server, the client promptly displays it on the screen.

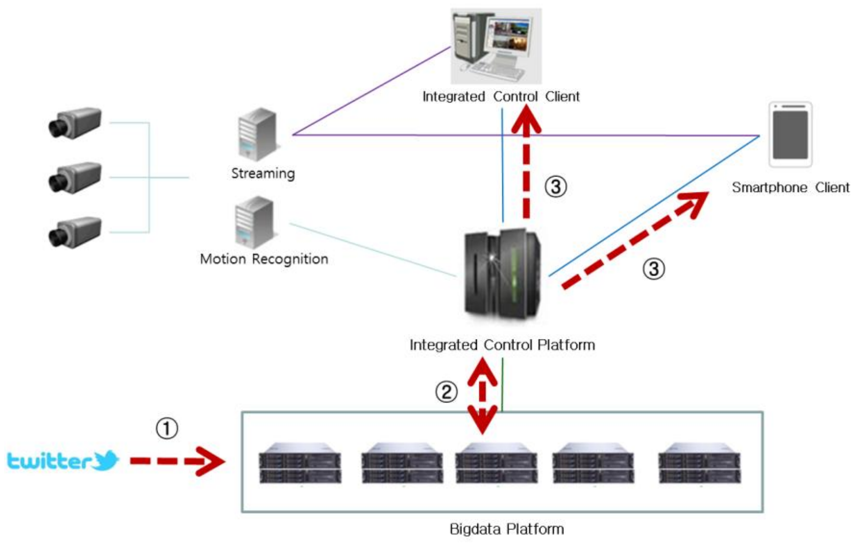

Fig. 6. Risk notification.

VI. Conclusion

This paper presents a monitoring and notification system based on risk analysis using map-reduce framework that can do risk prediction using big data. By using social media and access records of their users, the proposed system determines uncertain risks they could create. By using the risk information that has been obtained from the platform, it can determine whether they should be granted the requested rights to access to restricted areas. Also, it monitors some restricted areas through setting them up by the corresponding administrator and notifies him/her when and where the intrusion detection is activated. So, it will provide a control situation for the administrator in real time and can detect the risky moments that may occur in advance. Therefore it can reduce the probability of risk of security penetration. Also, since it receives the control situation and provides real-time control screen via smartphone, it is possible to improve the convenience of intrusion management in a very fast and effective way.

References

[1] C. Eaton, D. Deroos, T. Deutsch, G. Lapis, and P. C. Zikopoulos, *Understanding Big Data: Analytics for Enterprise Class Hadoop and Streaming Data*, Mc Graw-Hill Companies, 2012.

[2] R. D. Schneider, *Hadoop for Dummies Special Edition*, John Wiley&Sons Canada, 2012, ch. 1.

[3] D. Agrawal, S. Das, and A. E. Abbadi, "Big data and cloud computing: New wine or just new bottles?" *PVLDB*, vol. 3, no. 2, pp. 1647–1648, Sep. 2010.

[4] N. Marz and J. Warren, *Big Data: Principles and Best Practices of Scalable Realtime Data Systems*, Manning Publications, 2013.

[5] A. Bifet and E. Frank, "Sentiment knowledge discovery in Twitter streaming data," in *Proc. 13th International Conference on Discovery Science*, 2010, pp. 1-15.

[6] M. Tsytsarau and T. Palpanas, "Survey on mining subjective data on the web," *Data Mining and Knowledge Discovery*, vol. 24, no. 3, pp. 478-514, May 2012.

[7] H. Chen, R. Chiang, and V. Storey, "Business Intelligence and Analytics: From Big Data to Big Impact," *MIS Quarterly*, vol. 36, no. 4, pp. 1165-1188, Dec. 2012.

[8] J. Berthold, M. Dieterle, and R. Loogen, "Implementing Parallel Google Map-Reduce in Eden," in *Proc. Euro-Par*, LNCS 5704, 2009, pp. 990-1002.

[9] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107-113, Jan. 2008.

[10] J. Dean and S. Ghemawat, "MapReduce: a flexible data processing tool," *Communications of the ACM*, vol. 53, no. 1, pp. 72-77, Jan. 2010.

[11] Apache Hadoop project. [Online]. Available: http://hadoop.apache.org/

[12] R. A. S. Owen, T. Dunning, and E. Friedman, *Mahout in Action*, Manning Publications, pp. 225-357, 2010.

Yoon-Deuk Seo received his B.S. and M.S. degrees in computer science from Kyonggi University, Korea, in 2008 and 2010, respectively. He has been a Ph.D. student in the Department of Computer Science, Kyonggi University from 2010. His research interests include distributed computing, RFID systems, P2P networks and group communication.

Jinho Ahn received his B.S., M.S. and Ph.D. degrees in computer science and engineering from Korea University, Korea, in 1997, 1999 and 2003, respectively. He has been an associate professor in the Department of Computer Science, Kyonggi University. He has published more than 70 papers in refereed journals and conference proceedings and served as a program or organizing committee member or the session chair in several domestic/international conferences and editor-in-chief of Journal of Korean Institute of Information Technology and editorial board member of journal of Korean Society for Internet Information. His research interests include distributed computing, fault-tolerance, sensor networks and mobile agent systems.