Real-Time Network Anomaly Detection System Using Machine Learning

Shuai Zhao, Mayanka Chandrashekar, Yugyung Lee, Deep Medhi
Computer Science & Electrical Engineering Department, University of Missouri–Kansas City, USA
{szb53, mckw9, LeeYu, dmedhi}@mail.umkc.edu

***Abstract*—The ability to process, analyze, and evaluate real-time data and to identify their anomaly patterns is in response to realized increasing demands in various networking domains, such as corporations or academic networks. The challenge of developing a scalable, fault-tolerant and resilient monitoring system that can handle data in real-time and at a massive scale is nontrivial. We present a novel framework for real time network traffic anomaly detection using machine learning algorithms. The proposed prototype system uses existing big data processing frameworks such as Apache Hadoop, Apache Kafka, and Apache Storm in conjunction with machine learning techniques and tools. Our approach consists of a system for real-time processing and analysis of the real-time network-flow data collected from the campus-wide network at the University of Missouri–Kansas City. Furthermore, the network anomaly patterns were identified and evaluated using machine learning techniques. We present preliminary results on anomaly detection with the campus network data.**

I. Introduction

The structure and dynamic interactions in large network systems has become substantially complex. Traditional ways of real-time processing and analyzing of critical traffic in such networks may not be efficient in practice. In this work, we focus on building an efficient network management system that address real-time network flow-based data in an academic campus network. The flow data is generated sequentially, mostly with a timestamp, and also associated with multiple variables such as network traffic IPs, traffic throughput, and flow counts. It has been shown that flow-base network data can be well used for monitoring and the classification of the traffic patterns for either a persistent period of time or a continuous long period of time.

The Apache Hadoop system has become an important system for handling massive volumes of data [1]. However, this is not suitable for real-time applications. Recently, Apache Kafka [2] and Apache Storm [3] were introduced for big data processing to meet the growing need for real-time processing of streaming data. Apache Kafka is a distributed, scalable, publish-subscribe and fast messaging system. It offers high-throughput for streaming data and supports multiple data sources at the same time with load balancing and fault-tolerance. It might be well utilized if one has multiple large streaming data sources that need to be processed in a distributed and parallel fashion. In addition, data that is streamed into the Kafka framework can be saved for future use. Off-line detection algorithms [4], [5] can be applied to existing data. Apache Storm was designed to provide a framework for distributed, scalable and resilient computing for streaming data. The computing ability can be either extended by adding ad-hoc computing nodes or partitioned into different nodes. In this work, we combined Kafka and Storm for real time network anomaly detection for real time network-flow data.

This paper presents the design and implementation of a real time network anomaly detection system for network-flow data in a campus network. The rest of the paper is organized as follows. Related work is described in Section II. In Section III, we present our framework and explain various components. Preliminary study results are presented in Section IV. Finally, we conclude with a summary and future work in Section V.

II. Related Work

Cisco's NetFlow has become a de-facto standard of flow-based data protocol commonly used in enterprise networks. Broadly speaking, the anomaly analysis for flow-based data can be broken down into three steps [6]: 1) data collection, 2) data preprocessing, and 3) invocation of a detection algorithm. Previous research [7]–[9] focused on the analysis of network anomaly characteristics.

MINDS [10] proposed a near real-time analysis of flow data that presented a data analysis approach that can process NetFlow data every 10 minutes. The MINDS engine works in conjunction with the data analytic component to perform the batch processing, thus, it is not practically applicable to real-time network management.

Nfsen [11], Ntop [12] and Scrutinizer [13] are flow-based monitoring tools using a round-robin relational database. Their works were suitable when the majority of users' stream data was handled in a short time window, but it could not properly handle real-time query processing for large scale data analysis.

The recent work [14] proposed a real time anomaly detection framework based on Apache Storm that used machine learning algorithms, "k-NN" [5] and "Frequent Algorithm" [4], to find Top-N anomaly activities. It still does not appear to be capable of real-time analysis for multi-source streaming multiple data sources. It is mainly due to their offline data analysis processing. Distributed network measurement system [15] was implemented using a best effort in parallel and distributed Machine Learning (Mahout [16]). However, their work was not able to support real-time analysis due to Mahout's limitation built on top of the Hadoop HDFS file system.

978-1-4799-7795-6/15/$31.00 © 2015 IEEE

III. Proposed Architecture

A. Data Collection

The real network traffic of the campus network based using Cisco NetFlow [17] was collected at the campus data center at the University of Missouri–Kansas City (UMKC). Fig. 1 shows the campus network set up that consists of a data center (server farms) and four switches.

The UMKC campus network consists of four core switches inter-connected with each other using 1 *Gbps* link. Switch 1 and switch 3 have two 1 *Gbps* links for redundancy purpose. Switch 3 is the backbone switch for all other switches. It connects to the servers farms (Virtual Machine Servers at the campus data center) and Storage area network (SAN) through two 10 *Gbps* links. The number of NetFlow data generated from each switch is typically around 500,000 for each 5 minutes and varies significantly by the time of day.

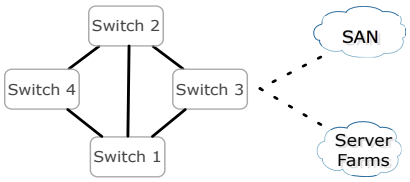


Fig. 1. UMKC data center core switches architecture

B. System Data Processing Architecture

Our proposed system is currently being developed using Apache Hadoop, Apache Storm, and Apache Kafka. Fig. 2 shows the architecture of our system. Apache's Hadoop distributed file system (HDFS) is an open source system for reliable, scalable, distributed computing. Hadoop is comprised of two major components, HDFS and MapReduce. For our real time streaming purpose, the HDFS component serves as a flow data producer.

Apache Kafka, a publish-subscribe messaging system, served as the central data backbone for distributed flow data. It handled different data sources by topics. For our purpose, Kafka handled the flow data as different topics for different switches. The Kafka's spouts can emit the flow tuple data into the Storm for further a anomaly detection process.

Apache Storm is an open source system for distributed real-time processing stream data, comparable to what Hadoop did for batch processing. Storm topologies consist of a graph of spouts (as data sources) and bolts (as data operations) as a form of stream groupings (for coordination). In our system, the Storm component handled various topologies for different anomaly detection situations.

This proposed system design provides the ability to combine both batch and real-time processing. In particular, the ability performs batch processing for the data saved by Kafka, HDFS, and real-time processing by analyzing each switch streaming flow data with Kafka and Storm.

Our Storm topology can be mapped to Kafka topics. Fig. 3 shows a topology setup for the anomaly detection in one of the switches. The streaming process is divided into four components: Kafka Spout, Data Preprocessing Bolt, Anomaly Detection Bolt, and Machine Learning Bolt.

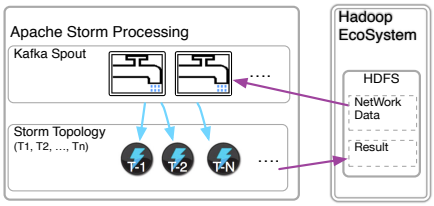


Fig. 2. Streaming architecture

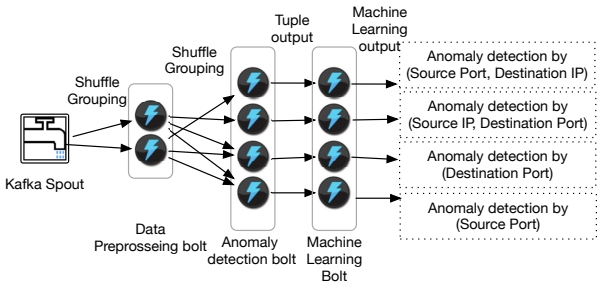


Fig. 3. Storm Topology

The Kafka Spout streamed the particular switch flow data into the Data Preprocessing Bolt and also saved the data for future use.

The Data Preprocessing Bolt performed a series of preprocessing operations on the flow data from the Kafka Spout, including data transformation and filtering. It analyzed the volumes of transferred bytes or packets of the NetFlow data and eliminated some trivial network traffic data (such as zero-byte flows) that may not be useful for anomaly detection of network traffic. Most of the zero-byte flows were used for handshaking of the TCP-IP connection [18].

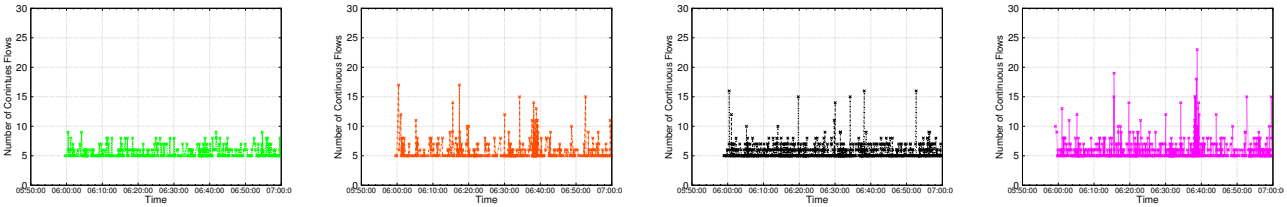
The Anomaly Detection Bolt implemented a anomaly detection mechanism based on previous work MINDS [10]. NetFlow version 5 data was collected from Section III-A and saved as flat file in every 5 minutes. We mainly considered four features from collected NetFlow data: Source and Destination IP, Source and Destination Port. We constructed a Connection-windows based feature table (See table I). Our detection mechanism reported if the same featured flow showed up over 5 times straight in last 5 seconds.

The Machine Learning Bolt conducted advanced anomaly detection over the anomaly traffic data detected by the Anomaly Detection Bolt. The data were used as training data for the machine learning process. The Machine Learning Bolt was implemented using a Weka Machine Learning tool [19] to improve the accuracy of anomaly detection tasks. The output was automatically saved and stored in HDFS for future use.

IV. Preliminary Experimental Results

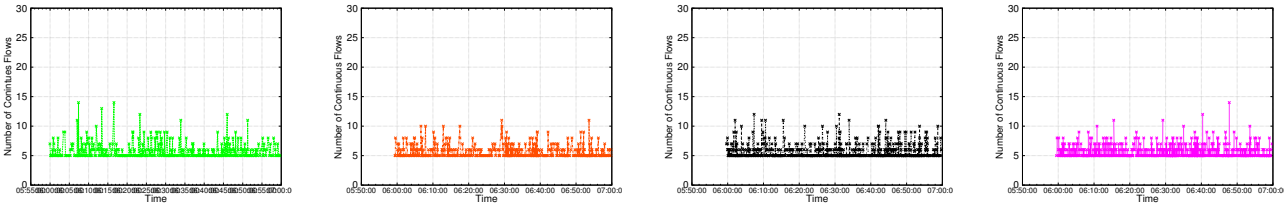
A. Experimental System Setup

Our experimental system setup includes 5 virtual machines using Virtualbox. All the nodes have the same configuration setup with 4-core of Intel-i7 3.2Ghz, 10GB of RAM. The connection between each node is 1 *Gbps*. Fig. 8 shows the application deployment on each node. The current system is set up with Hadoop YARN (2.5.1) on CentOS 6.5 as a single



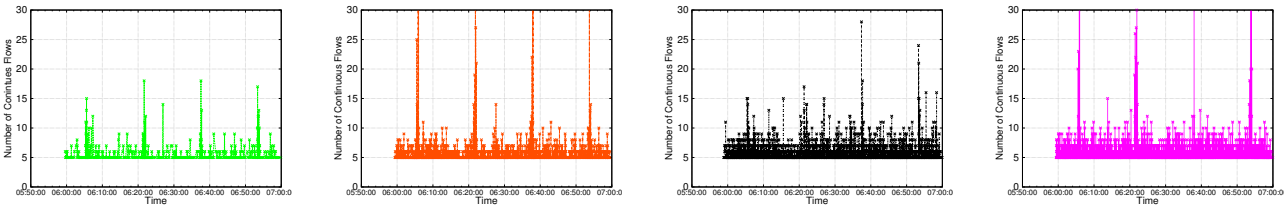
(a) (Source Port, Destination IP) (b) (Source IP, Destination Port) (c) (Unique Destination IP) (d) (Unique Source IP)

Fig. 4. Real time anomaly detection for switch 1: 11/05/2014 06:00-07:00



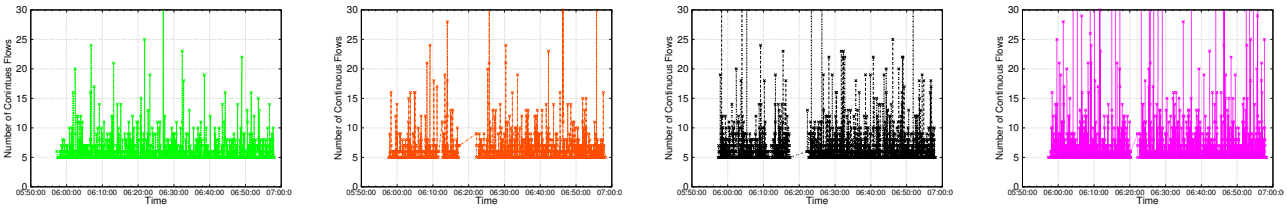
(a) (Source Port, Destination IP) (b) (Source IP, Destination Port) (c) (Unique Destination IP) (d) (Unique Source IP)

Fig. 5. Real time anomaly detection for switch 2: 11/05/2014 06:00-07:00



(a) (Source Port, Destination IP) (b) (Source IP, Destination Port) (c) (Unique Destination IP) (d) (Unique Source IP)

Fig. 6. Real time anomaly detection for switch 3: 11/05/2014 06:00-07:00



(a) (Source Port, Destination IP) (b) (Source IP, Destination Port) (c) (Unique Destination IP) (d) (Unique Source IP)

Fig. 7. Real time anomaly detection for switch 4: 11/05/2014 06:00-07:00

TABLE I
ANOMALY DETECTION FEATURES FOR MACHINE LEARNING

Feature	Feature Details
A	(Source Port, Destination IP)
B	(Source IP, Destination Port)
C	(Unique Destination IP)
D	(Unique Source IP)

cluster. Each cluster is configured with a single NameNode and three DataNodes in conjunction with a Zookeeper server for coordination.

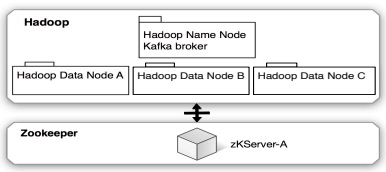


Fig. 8. Experimental system setup

B. Real-time anomaly detection

Fig. 4 - Fig. 7 show the results of anomaly detection at all switches by streaming one hours' real time NetFlow data with the feature listed in Table I. Switches 1 to 3 have a relatively small set of flow data marked for each of the features. Switch 4 shows dramatically different traffic behaviors compared to the other switches. We conducted a manual evaluation of our results with several campus network experts at UMKC's data center and learned that that the flows in Switches 1, 2, and 3 are considered to be normal network activities. They were identified as typical activities for campus internal applications such as the logging system, multicasting traffic, and data center utility services. In particular, we were surprised to see a significant amount of repeated traffic for a short period of time. Switch 4 had large amounts of data on traffic for external applications. The results show large amounts of traffic from a Source IP address to one of UMKC's data center servers. From the time-frequency analysis on end-to-end traffic, we

found that the frequency of the traffic between these two switches was 175 times in a short duration Similarly, 283 times from another Source IP to another UMKC data center server in a short duration of time. From the manual evaluation for anomaly detection, we conclude that the network anomaly activities can be analyzed and detected in real time. We plan to conduct an in-depth analysis of the traffic anomaly detection with any abnormal events and enhance our proposed system.

C. Machine Learning for Network Anomaly Detection

In this section, we present the network anomaly detection approach based on machine learning (ML) techniques. For this purpose, we investigated three ML classification algorithms including Naïve Bayesian (NB), Support vector machine (SVM), and Decision tree (DT). These are the most widely used supervised learning techniques while simultaneously achieving high-accuracy performance. For accuracy validation for these predictive models, 10 fold cross-validation technique was used. The features are constructed using the connection-window based features shown in Table I. These features were combined into two groups such as A, B and C, D for each switch. Table II shows the results for each switch based on these three different predictive algorithms (NB, SVM, DT). Overall, the classification accuracies with the feature group C, D show high accuracy in comparison to the feature group A, B. The classification with switch 3 data shows higher accuracy compared to the others. SVM shows the highest accuracy (99.90%) with the feature group C, D of switch 3 and the lowest accuracy (70.63%) with features A, B of switch 1. The reason why it gets lower accuracy could be due to traffic size is being lower on switch 1. The ML-based Network Anomaly Detection approach using these predictive models gives promising results with an average of 90.3%.

TABLE II
ACCURACY RESULT

Switch Name	Features	Classifiers	Accuracy %
Switch 1	A + B	NB	71.46%
		SVM	70.63%
		DT	72.02%
	C + D	NB	93.97%
		SVM	98.79%‘
		DT	95.78%
Switch 2	A + B	NB	81.19%
		SVM	77.85%
		DT	81.42%
	C + D	NB	99.18%
		SVM	99.18%
		DT	99.18%
Switch 3	A + B	NB	93.24%
		SVM	95.6%
		DT	95.10%
	C + D	NB	99.03%
		SVM	99.90%
		DT	99.85%
Switch 4	A + B	NB	84.14%
		SVM	80.97%
		DT	83.7%
	C + D	NB	99.3%
		SVM	99.07%
		DT	96.75%

V. CONCLUSION AND FUTURE WORK

In this preliminary work, we present a novel real-time system for network anomaly detection by utilizing state-of-the-art approaches including Apache Storm, Apache Kafka and applying real-time analytics on streaming data for network monitoring and management. We obtained promising preliminary results for real-time network anomaly detection. Future research will be on in-depth analysis of anomaly detection for a real-time network management system, enhancement of machine learning and optimization algorithms for real-time processing and high accuracy, and implementation of visualizing tools for comprehensive understanding of dynamic behaviors of complex networks.

ACKNOWLEDGMENT

We thank Jim Schonemann and Frank Magrone of UMKC Campus Information Services division for numerous discussions about the campus data and potential anomalies. This work is supported in part by the National Science Foundation grant # CNS-1217736.

REFERENCES

[1] J. Dean and S. Ghemawat, “Mapreduce: simplified data processing on large clusters,” *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.
[2] J. Kreps, N. Narkhede, J. Rao *et al.*, “Kafka: A distributed messaging system for log processing,” in *Proceedings of 6th International Workshop on Networking Meets Databases (NetDB), Athens, Greece*, 2011.
[3] A. Toshniwal, S. Taneja, A. Shukla, K. Ramasamy, J. M. Patel, S. Kulkarni, J. Jackson, K. Gade, M. Fu, J. Donham, and other, “Storm@ twitter,” in *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, 2014.
[4] R. M. Karp, S. Shenker, and C. H. Papadimitriou, “A simple algorithm for finding frequent elements in streams and bags,” *ACM Transactions on Database Systems (TODS)*, vol. 28, no. 1, pp. 51–55, 2003.
[5] C. Wang, I. A. Rayan, and K. Schwan, “Faster, larger, easier: reining real-time big data processing in cloud,” in *Proceedings of the Posters and Demo Track*, 2012.
[6] G. Munz and G. Carle, “Real-time analysis of flow data for network attack detection,” in *Integrated Network Management, 2007. IM’07. 10th IFIP/IEEE International Symposium on*. IEEE, 2007, pp. 100–108.
[7] P. Barford and D. Plonka, “Characteristics of network traffic flow anomalies,” in *Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement*. ACM, 2001, pp. 69–73.
[8] F. Dressler and G. Munz, “Flexible flow aggregation for adaptive network monitoring,” in *Local Computer Networks, Proceedings 2006 31st IEEE Conference on*. IEEE, 2006, pp. 702–709.
[9] C. Gates and D. Becknel, “Host anomalies from network data,” in *Information Assurance Workshop, 2005. IAW’05. Proceedings from the Sixth Annual IEEE SMC*. IEEE, 2005, pp. 325–332.
[10] L. Ertoz, E. Eilertson, A. Lazarevic, P.-N. Tan, V. Kumar, J. Srivastava, and P. Dokas, “Minds-minnesota intrusion detection system,” *Next Generation Data Mining*, pp. 199–218, 2004.
[11] NfSen. [Online]. Available: http://nfsen.sourceforge.net
[12] Ntop. [Online]. Available: http://www.ntop.org
[13] P. Scrutizer. [Online]. Available: http://www.plixer.com
[14] Y. Du, J. Liu, F. Liu, and L. Chen, “A real-time anomalies detection system based on streaming technology,” in *Intelligent Human-Machine Systems and Cybernetics (IHMSC), 2014 Sixth International Conference on*, vol. 2. IEEE, 2014, pp. 275–279.
[15] Q. Zhang, Y. Jin, Y. Cui, and M. Song, “A distributed network measurement system based on hadoop,” in *Wireless Communications, Networking and Mobile Computing (WiCOM), 2012 8th International Conference on*. IEEE, 2012.
[16] A. Mahout. [Online]. Available: http://mahout.apache.org
[17] B. Claise, “Cisco systems netflow services export version 9,” 2004.
[18] S. Zhao, K. Leftwich, M. Owens, F. Magrone, J. Schonemann, B. Anderson, and D. Medhi, “I-can-mama: Integrated campus network monitoring and management,” in *Network Operations and Management Symposium (NOMS), 2014 IEEE*. IEEE, 2014, pp. 1–7.
[19] G. Holmes, A. Donkin, and I. H. Witten, “Weka: A machine learning workbench,” in *Intelligent Information Systems, 1994. Proceedings of the 1994 Second Australian and New Zealand Conference on*, 1994.