Volume 3, Issue 12, December 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Anomaly or Inconsistency Extraction Using Improved FP-Growth Algorithm

Pravin Gaikwad
PG Student of Department of
Computer Engineering, SCOE, Pune,India

Jyoti Kulkarni
Assistant Professor of Department of
Computer Engineering, SCOE, Pune, India

Abstract*— identifying network anomalies is essential in enterprise and provider networks for diagnosing events, like attacks or failures. Anomaly extraction method refers to automatically finding, in a large set of flows observed during an anomalous time interval, the flow associated with the anomalous events. It is important for root-cause analysis, attack mitigation, network forensics and anomaly modelling. In this paper, we use meta-data provided by several histogram-based detectors to identify suspicious flows, and then apply association rule like Improved FP-Growth Algorithm to find and summarize anomalous flows. Using rich traffic data from a network, we show that our technique effectively finds the flows associated with anomalous events in all studied cases. In addition our algorithm shows a very small number of false positive rates and executes fast than Apriori and FP-Growth algorithm for large data sets.

Keywords*— Computer network, data mining, detection algorithm, association rules, Improved FP algorithm, compound single linked list.

I. INTRODUCTION

A. Motivation

Detecting and identifying network anomalies are becoming an important factor to consider when taking care of network security, uptime and performance of ISPs and large scale networks. An anomaly is defined as a "Deviation or departure from the normal or common order, form, or rule". Anomaly detection techniques are last line of defence when other approach fails to detect security threats or other problem. They have pose number of interesting research problems, involving statistics, modelling, and efficient data structures. Nevertheless, they have not yet gained widespread adaptation, as a number of challenges, like reducing number of false positive rate and calibration, remain to be solved.
In this paper, we are interested in the problem of identifying the traffic flows associated with an anomaly during a time interval with an alarm. We call finding these flows the anomalous flow extraction or anomaly extraction. Anomaly extraction reflects the goal of gaining more information about an anomaly alarm. Identified anomalous flows can be used for a number of applications, like root-cause analysis of the event causing an anomaly, network forensics, improving anomaly detection accuracy, and anomaly modelling.

B. Anomaly Extraction

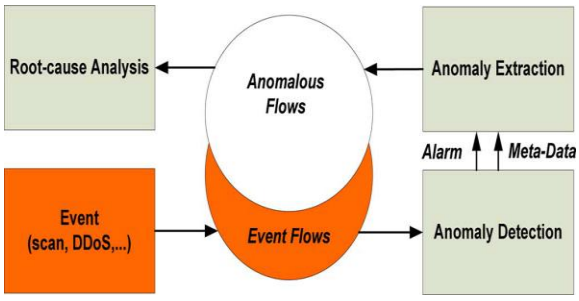

Fig 1: High level goal of anomaly extraction is to filter and summarize the set of anomalous flows that coincide with the flows caused by a network event such as denial-of-service (DoS) attack.

In Fig. 1, we present a high level goal of anomaly extraction. In the bottom of the figure, events with network level footprint, like attack or failure, trigger event flow, which after analysis by an anomaly detector, may raise an alarm. Ideally, we would like to extract exactly all triggered event flows. The goal of anomaly extraction process is to find a set of anomalous flows coinciding with event flows. An anomaly detection system may provide a meta-data relevant to an alarm that helps to find the set of candidate anomalous flows. For example, anomaly detection systems analysing histograms may indicate histogram bins than an anomaly affected, e.g., a range of IP addresses and Port numbers. Such

meta-data can be used to restrict the candidate anomalous flows to these that have an affected IP address and port numbers. To extract an anomalous flow from network traffic, one could build a model describing normal flow characteristic and use the model to identify deviating flows. To build such model is very challenging issue due to wide variability of flows characteristic. Similarly, one could compare flows during an interval with flow from normal or past interval to the new flows or flow with significant increase/decrease in their volume to search for changes or to identify new flows that were not previously observed [6] [7]. Such approaches essentially perform anomaly detection at the level of individual flows and could be used to identify anomalous flows.

C. Contributions

In this paper, we take an alternative approach to identify anomalous flows that combines and consolidate information from multiple histogram-based detectors and an Improved FP-growth algorithm is applied to speed up the system rate and decrease the false positive rate as compared to Apriori and FP-growth algorithm [2]. Compared to other possible approaches, our method does not rely on past data for normal interval or normal models. Intuitively, each histogram-based detectors provide and additional view of network traffic.

A detector may provide a set of candidate anomalous flows. The main reason of applying an association rule is that *Anomalies typically result in many flows with similar characteristic*, e.g., common IP address or port numbers, since they have a common root-cause, like network failure or a scripted denial-of-service (DoS) attack.

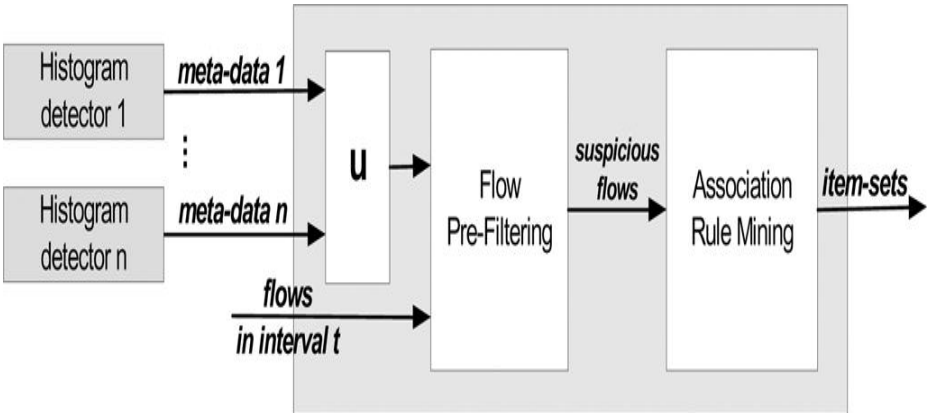

Fig 2: Overview of our approach to the anomaly extraction problem. The fig illustrate how meta-data for filtering flows is consolidate from *n* traffic features by taking the union and how suspicious flows are pre-filtered and anomalous flows are summarized in item-sets by association rule mining.

D. Outline

The rest of this paper is structured as follows. Section II describes our technique for extracting anomalous traffic and a new algorithm called Improved FP-growth algorithm based on compound single linked list to overcome the disadvantages of Apriori and FP-growth algorithm. Section III describes a related work of our system; Finally, Section IV concludes our paper.

II. METHODOLOGY

An overview of our approach to the anomaly extraction is given in fig.2. An *n* number of histogram-based anomaly detectors monitors the network traffic and detect anomalies in an online fashion [1]. Upon detecting anomalies, we use a union set of meta-data provided by *n* histograms-based detectors to pre-filter a set of suspicious flows. This pre-filtering process is necessary in order to eliminate a large part of normal flows. Then association rules are applied in order to generate frequent item-sets in the set of suspicious flows [1]. The basic assumption behind applying association rules is that frequent item-sets in the pre-filtered data are often related to anomalous event. The association algorithm like Improved FP-growth algorithm is applied to overcome the disadvantages of Apriori and FP-growth. Improved FP-growth improves the algorithm both in runtime and main memory consumption, which does its work without any complex data structure. The entire anomaly extraction process is automated and can take place both in online and offline fashion.

A. Flow Pre-filtering

Pre-filtering process usually removes a large part of the normal traffic. This process is desirable for two reasons. 1] It generates a substantially smaller data dataset that results in faster in the following steps. 2] It improves the accuracy of association rule mining by removing flows that could result in false-positive item-sets. Assume a time interval *t* with an anomaly. Pre-filtering selects all flows that match the *union* of the meta-data provided by *n* histogram-based detectors.

An important detail of our approach is that we keep flows matching any of the meta-data instead of flow matching *all* the meta-data provided by *n* detectors. We take *union* of the flows matching meta-data rather than *intersection* of the flows matching meta-data. Taking the union is important because identified meta-data can be flow disjoint, meaning that they appear in different flows, in which case the intersection is empty. The intersection of the flows matching the meta-data

would be empty, whereas the union would include anomalous flows. It shows that the union results in fewer false positive that the intersection which may miss entire anomaly.

B. Frequent Item-Sets Mining

Association rule plays an important part in the field of data mining, which is used to mine the association rules in a given data set and is put forward by R.Agraawal [4] first. It divides the mining process into two steps: (1) find all frequent itemsets. (2) generate strong association rules from the frequent itemsets. The main purpose of applying association rules is that to find frequent item-sets to extract anomalous flows from large data-sets in time interval. Our assumption for applying frequent item-set mining to anomaly extraction problem is that *anomalies typically result in large number of flows with similar characteristic*. e.g., IP address, port numbers, since they have a common root cause analysis like network failure or scripted DoS attack.

The transaction width is defined as the number of items present in a transaction. Each transaction has a width of five since each flow record has five associated feature corresponding to its srcIP, dstIP, srcPort, dstPort, #packets. For example the item i_1 ={srcPort : 80} refers to a source port number equal to 80, while item i_2 = { dstPort : 80} refers to destination port number 80, a transaction cannot have to items same feature type.

THE IMPROVED FP-GROWTH ALGORITHM BASED ON COMPOUND SINGLE LINKED LIST

1] The steps to construct the compound single linked list

1. The first scan of database is the same as the FP-tree. The scan of the database derives the set of frequent items (1-itemsets) and their support counts (frequencies). The set of frequent items is sorted in the order of descending support count. This resulting set or list is denoted L and an item header table is built.
2. The second scan of database is different from the FPgrowth. It is processing the items in each transaction in L order, and then inserting the items in each transaction into the single linked list recursively. The items' order in each single linked list is according to L order.

2] The realization of the pseudocode of the improved algorithm

1. The storing data structure is consist of three parts-

i] Header Table.
Item-name, support count, node link

ii] Frequent Item-sets.
Frequent item, Pointer.

iii] Single Linked List.
Item, Count, Pointer.

2. The realization of the pseudocode:

Input: A transaction database, D; minimum support threshold, min_sup
Output: The collection of the frequent pattern
Method：
(1)The first scan of the database derives the set of frequent items (1-itemsets) and their support counts (frequencies). The set of frequent items is sorted in the order of descending support count. This resulting set or list is denoted L and an item header table is built.
(2)Scan database a second time. The items in each transaction are reserved and processed in L order, then stored in the single linked table Frequent itemsets.
(3)// i is the sequence of each transaction, p is the pointer pointing to one of the frequent itemsets in each trasaction, q is the total number of the transaction.

```
        for (i=1; i≤q; i++)
        do { scan number i's trasaction
           let p point to number I's first frequent itemsets;
           while (p≠Null)
        { if (p→next==Null) end;
        else { find p's pointing frequent item in header table
        traverse the single linked table that p→next points to
        during the traversal ;
if (the current node is in the corresponding single linked table)
incremented by 1
        else { generate a new node and insert it to single linked
table according to the order of L, the count is assigned by 1;
        }
        }
p= p→next
    }
}
```

C. Histogram-Based Detectors

Histogram-based anomaly detectors [8] [9] [10] [11] have been work well for detecting anomalous behaviour and changes in traffic distribution. Our histogram-based detector uses the Kullback-Leibler (KL) distance to detect anomalies. The KL distance has been successfully applied for anomaly detection in previous work [12] [11]. Each histogram-based detector monitors a flow feature distribution, like distribution of IP address and Port numbers. We assume that for *n* histogram detectors of *n* traffic features *m* histogram bins are constructed.

During a time interval *t,* an anomaly detector module constructs histogram for number of flows per traffic feature. At the end of interval, it computes for each histogram the KL distance between the distribution of current interval and reference interval distribution. The KL distance of given discrete distribution *q* to reference distribution *p* be

$$D\,(p//q) = \sum_{i=0}^{m} p_i \log(p_i/q_i).$$

D. Histogram Cloning and Voting

Histogram cloning is a promising technique applied to Histogram-Based Detection. The motivation behind it is to maintain multiple randomized histograms (of the same feature), hence it will obtain additional views of network traffic. This technique is realized in [13] by creating n histogram-based detectors corresponding to *n* different traffic features. For each of the *n* features there are *m* bins per time interval, and by applying a hash function to each of the clones, one makes sure that each feature value is placed randomly into one of the *m* bins. This differs from classical binning, which tends to place adjacent feature values (e.g. source ports) next to each other in a histogram.

E. Parameter Estimation

The various parameters associated with our approach, they are as follows.

TABLE I
PARAMETERS INCLUDING DESCRIPTION USED IN SECTION II

Parameter	Description
n	**Number of detectors**
w	**Interval length**
m	**Hash Function Length**
k	**Number of clones**
l	**Voting Parameter**
s	**Minimum support**

Number of Detectors n: In this paper, we use five detectors, which correspond to five features that are frequently used for network traffic anomaly detection: source IP addresses, destination IP addresses, source port numbers, destination port numbers, and number of packets. In principle, if computational overhead is reasonable, you can also use more features or detectors to get a additional view of network anomalies. Other features are like, the number of packets per flow, the average packet size, the duration of a flow, the source/destination autonomous system (AS) numbers, and the geographical distribution of IP addresses.

Interval Length w: The interval length determines the detectable anomaly scale, i.e., it becomes harder to detect short disruptions that contain only few flows with longer intervals .On the other hand, it is not always desirable to detect such short disruptions. Hence, the desired number of daily or weekly anomalous alarms can be used to set the interval length. One last implication is that a larger results in more flows to be processed by association rule mining and in higher computational overhead. Nevertheless, the overhead of association rule mining after pre-filtering is relatively low.

Hash Function Length m: The function length is also involved in detection sensitivity. The smaller the length of hash function more flows are aggregated per hash function bin. The, *m* should be selected together with *w* based on a desired number of daily/weekly anomalous alarms. Smaller bins are preferable for anomaly extraction.

Voting Parameter l and k: The parameter *k* determines the total number of histogram clones used. The computational requirements in terms of memory and CPU scale are linearly with *k*. The parameter *l* determines the lower bound for number of clones that need to select a feature value to be included in the final meta-data. The parameter *l* impacts the number of flows selected in the pre-filtering step and thus accuracy of our approach.

Minimum Support s: The parameter *s* determines the frequency threshold above which an item-set is extracted by Enhanced FP as a possible set of anomalous flows. A large *s* extracts no or few item-sets, which were always associated with anomalous events. On other hand small value of *s* results in more item-sets and in a small but higher rate of false positives. The value of *s* needs to be determining by trial and error.

III. RELATED WORK

Most related to our work, Silveira and Diot [13] recently introduced a tool called URCA (Unsupervised Root Cause Analysis) that searches anomalous flows by iteratively eliminating normal flows. Ding Zhenguo, Wei Qinqin [2] introduced a algorithm which shows the simplicity and fast processing of advanced FP algorithm. D. Brauckhoff and X. Dimitropoulos [1] is a automated anomaly extraction system which extract features is going to affect the anomaly. It is important for root cause analysis, network forensics, attack mitigation, and anomaly modelling. DoWitcher [17] is a scalable system for worm detection in backbone network. Part of this system automatically constructs a flow-filter mask

from the intersection of suspicious attributes provided by different detectors. In addition, the authors discuss how their method can be used for anomaly extraction. However, the work and evaluation focus primarily on anomaly detection.
Association rules have been successfully applied to different problems on networking. Lee and Stolfo [18] show how association rules can be used to extract interesting intrusion pattern. Vaarandi [19] provides an optimized implementation of Apriori and demonstrate how it can be used for summarized traffic flow records. C. Borgelt [5] shows a advantages of FP-growth algorithm and false positive rate and number a candidate item-sets.

IV. CONCLUSION

Anomaly extraction takes as input a large set of flows and aims at finding the flows associated with the event that triggered an observed anomaly. It is very useful for finding the root cause of detecting anomalies, which helps in attack mitigation, anomaly modelling and network forensics. Optimizing the scalability and efficiency of frequent item-set mining for dealing with big network data including stream processing is one open problem. To avoid such problems of big network data, we introduced Improved-FP algorithm, The improved FP-growth is mined in one direction, using the header table in the original FP-tree, and a compound single linked list is constructed. The result shows that the improved algorithm is better than the FP-growth. By comparing these frequent item-set mining algorithms apriori and fp-growth and Improved FP, Improved FP is faster and requires less memory.

ACKNOWLEDGMENT

I am very thankful to my guide for guiding me and heartly thankful to IJARCSSE to give me such a wonderful chance for publishing my paper. Also thankful to Microsoft to help me in writing this paper.

REFERENCES

[1] D. Brauckhoff, X. Dimitropoulos, A. Wagner, and K. Salamatian, "Anomaly extraction in backbone networks using association rules,"in *Proc. IEEE ACM TRANSACTION ON NETWORKING*, VOL.20. NO 6, DECEMBER 2012.

[2] Ding Zhenguo, Wei Qinqin, Ding Xianhua "An Improved FP-growth Algorithm Based on Compound Single Linked List". In *Proc.* IEEE 2009.

[3] W. A. Kosters,W. Pijls, and V. Popova, "Complexity analysis of depth first and FP-growth implementations of APRIORI," in *Proc. MLDM*, 2003, pp. 284–292.

[4] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules in large databases," in *Proc. 20th VLDB*, Santiago de Chile, Chile, Sep. 12–15, 1994, pp. 487–499.

[5] C. Borgelt, " An implementation of the FP-growth Algorithm", Otto-von-Guericke-University of Magdeburg.

[6] B. Krishnamurthy, S. Sen, Y. Zhang, and Y. Chen, "Sketch-based change detection: Methods, evaluation, and applications," in *Proc. 3rd ACM SIGCOMM IMC*, 2003, pp. 234–247.

[7] G. Cormode and S. Muthukrishnan, "What's new: Finding significant differences in network data streams," *IEEE/ACM Trans. Netw.*, vol. 13, no. 6, pp. 1219–1232, Dec. 2005.

[8] A. Kind, M. P. Stoecklin, and X. Dimitropoulos, "Histogram-based traffic anomaly detection," *IEEE Trans. Netw. Service Manage.*, vol. 6, no. 2, pp. 110–121, Jun. 2009.

[9] M. P. Stoecklin, J.-Y. L. Boudec, and A. Kind, "A two-layered anomaly detection technique based on multi-modal flow behaviormodels," in *Proc. 9th PAM*, 2008, Lecture Notes in Computer Science, pp. 212–221.

[10] X. Li, F. Bian, M. Crovella, C. Diot, R. Govindan, G. Iannaccone, and A. Lakhina, "Detection and identification of network anomalies using sketch subspaces," in *Proc. 6th ACM SIGCOMM IMC*, 2006, pp. 147–152.

[11] K. H. Ramah, K. Salamatian, and F. Kamoun, "Scan surveillance inInternet networks," in *Proc. Netw.*, 2009, pp. 614–625.

[12] Y. Gu, A. McCallum, and D. Towsley, "Detecting anomalies in network traffic using maximum entropy estimation," in *Proc. 5th ACM SIGCOMM IMC*, 2005, pp. 32–32.

[13] F. Silveira and C. Diot, "URCA: Pulling out anomalies by their root causes," in *Proc. IEEE INFOCOM*, Mar. 2010, pp. 1–9.

[14] M. Zaki, "Scalable algorithms for association mining," *IEEE Trans. Knowl. Data Eng.*, vol. 12, no. 3, pp. 372–390, May–Jun. 2000.

[15] A. Wagner and B. Plattner, "Entropy based worm and anomaly detection in fast IP networks," in *Proc. 14th IEEE WETICE*, 2005, pp. 172–177.

[16] M. V. Mahoney and P. K. Chan, "Learning rules for anomaly detection of hostile network traffic," in *Proc. 3rd IEEE ICDM*, 2003, pp.601–604.

[17] S. Ranjan, S. Shah, A. Nucci, M. M. Munafò, R. L. Cruz, and S. M. Muthukrishnan, "Dowitcher: Effective worm detection and containment in the Internet core," in *Proc. IEEE INFOCOM*, 2007, pp. 2541–254.

[18] W. Lee and S. J. Stolfo, "Data mining approaches for intrusion detection," in *Proc. 7th USENIX Security Symp.*, 1998, vol. 7, p. 6.

[19] R. Vaarandi, "Mining event logs with SLCT and LogHound," in *Proc. IEEE NOMS*, Apr. 2008, pp. 1071–1074.

