Multi way TRIE Based Data Structure for Retrival of Time Series Traffic with Varying Granularity

Akhil M R
Department of Computer Science & Engineering
Mar Athanasius College of Engineering Kothamangalam, Kerala

Rahul V Cheeran
Department of Computer Science & Engineering
Mar Athanasius College of Engineering Kothamangalam, Kerala

Joby George
Department of Computer Science & Engineering
Mar Athanasius College of Engineering Kothamangalam, Kerala

Abstract

Analyzing the network traffic for network management require the fine grained information regarding the network traffic. In real-time, analyzing the traffic is a tedious task since there is no dedicated data structure to handle the traffic data. To obtain the fine grained information on the traffic it requires a large overhead. In this method the traffic information is stored at each observer point and the required information is passed to the requesting network manager. Since the network manager receives only the intended information the network won't get flooded with the log information. As a dedicated data structure for analyzing traffic, we propose a three tree structure consisting of Time Tree, Source Tree and Destination Tree. Time tree hold data regarding the time slots and the Source & Destination Tree holds IP addresses and packet information. Source and Destination Tree is constructed using Multiway Tries. A wide range of granularity over the time and IP addresses are possible in the system.

Keywords: Time Tree, Source IP Tree, Destination IP Tree, Traffic data, Traffic Observer

I. INTRODUCTION

Internet is the most widely used technology for information exchange. The reach of internet had become far beyond the expectation. About 45 % of the total world population uses internet now a days. Internet traffic will likely more than double in next five years, and much of it will be used for online video and social-media usage. To manage internet traffic the time series traffic details are essential. Traffic engineering is branch that deals with network traffic handling. Analyzing the traffic data is a tedious task in current scenario.

To improve the performance of network dynamic routing is necessary, to enable dynamic routing traffic data consisting of small time granularity is required. Dynamic routing, also known as adaptive routing, is the ability of a system, in which routes are characterize by their destination, to change the path that the route takes through the system in response to a adapt to situation. In dynamic routing method the servers collects traffic data periodically from router component and analyze the time series of network traffic. But such a practice is not efficient since it makes a large overhead of communication. One of the alternative methods to overcome the overheads is by transferring only the required traffic information according to the request. Above described method works efficiently in dynamic routing since dynamic routing requires traffic data of aggregated flows, which passes through same nodes, instead of traffic data of each flow, while aggregated flow change in time.

To improve the performance of network dynamic routing is necessary, to enable dynamic routing traffic data consisting of small time granularity is required. Dynamic routing, also known as adaptive routing, is the ability of a system, in which routes are characterize by their destination, to change the path that the route takes through the system in response to a adapt to situation. In dynamic routing method the servers collects traffic data periodically from router component and analyze the time series of network traffic. But such a practice is not efficient since it makes a large overhead of communication.

One of the alternative methods to overcome the overheads is by transferring only the required traffic information according to the request. Above described method works efficiently in dynamic routing since dynamic routing requires traffic data of aggregated flows, which passes through same nodes, instead of traffic data of each flow, while aggregated flow change in time. Here in this paper, we propose a method that enables retrieval of the data regarding the time series of network traffic with the requisite granularity that changes in time. Traffic observers are computers that analyze traffic data of connected routers. In this method, traffic observers are deployed at each monitoring point. The traffic observer stores all traffic data from the linked or nearby routers. If traffic data is requested, the manager sends a call including the required granularity to the traffic observers unit. Then, each traffic observer analyzes the stored data to construct the traffic information with the required granularity, and

sends back the constructed information. By using this method, the overhead to collect the traffic information can be reduced by collecting only the required traffic information with the required granularity.

In the proposed method the way in which traffic data has been stored has a great influence. If we simply store each fine-grained traffic data, it takes creates large overheads to retrieve the coarse grained data, to obtain the traffic amount of an IP address prefix, the traffic observer is required to search all traffic data matching the IP address prefix. So an efficient data structure to store the traffic data has to build. The data structure should be capable of fast insertion and immediate retrieval of network traffic. Which means the proposed data structure should capable of storing the time series of traffic data in such way that immediate retrieval is possible for a wide set of prefixed granularity level.

The data structure which we propose consists of three tree structures. Each tree is used for storing one of the required components of time series network traffic data. First tree stores the time component of traffic data where the second and third tree stores the Source and Destination IP's of the traffic data respectively. Each node has a reference to the pointer of the tree representing the next field of traffic data. Moreover we built the IP trees by using Multiway Tire structure which reduces the time for insertion and retrieval of traffic data from the data structure.

II. RELATED STUDIES

A. SFLOW:

As mentioned in " InMon Corporation's sFlow: A Method for Monitoring Traffic in Switched and Routed Networks " by P. Phaal, S. Panchen, and N. McKee , sFlow is a technology for monitoring traffic in data networks containing switches and routers. In particular, it defines the sampling mechanisms implemented in an sFlow Agent for monitoring traffic, the sFlow MIB for controlling the sFlow Agent, and the format of sample data used by the sFlow Agent when forwarding data to a central data collector. The sFlow monitoring system consists of an sFlow Agent (embedded in a switch or router or in a standalone probe) and a central data collector, or sFlow Analyzer. The sFlow Agent uses sampling technology to capture traffic statistics from the device it is monitoring. sFlow Datagrams are used to immediately forward the sampled traffic statistics to an sFlow Analyzer for analysis.

B. Netflow:

As mentioned in "Cisco Systems NetFlow Services Export Version 9 "by B. Claise. A flow is defined as a unidirectional sequence of packets with some common properties that pass through a network device. These collected flows are exported to an external device, the NetFlow collector. Network flows are highly granular; for example, flow records include details such as IP addresses, packet and byte counts, timestamps, Type of Service (ToS), application ports, input and output interfaces, etc. Exported NetFlow data is used for a variety of purposes, including enterprise accounting and departmental charge backs, ISP billing, data warehousing, network monitoring, capacity planning, application monitoring and profiling, user monitoring and profiling, security analysis, and data mining for marketing purposes.

C. Dynamic Routing:

Advances in the technology of modern telecommunication systems have led to considerable interest in schemes which can dynamically control the routing of calls within a network. The purpose of such dynamic routing schemes is to adjust routing patterns within the network in accordance with varying and uncertain offered traffics, to make better use of spare capacity in the network resulting from dimensioning upgrades or forecasting errors, and to provide extra flexibility and robustness to respond to failures or overloads.

D. Traffic Measurement with Granularity:

The idea proposed by L. Yuan, C.-N. Chuah, and P. Mohapatra in the paper "ProgME: Towards Programmable Network Measurement". Traffic measurements provide critical input for a wide range of network management applications, including traffic engineering, accounting, and security analysis. Existing measurement tools collect traffic statistics based on some predetermined, inflexible concept of "flows." They do not have sufficient built-in intelligence to understand the application requirements or adapt to the traffic conditions. Consequently, they have limited scalability with respect to the number of flows and the heterogeneity of monitoring applications. This paper presents ProgME, a Programmable MEasurement architecture based on a novel concept of flowset-an arbitrary set of flows defined according to application requirements and/or traffic conditions. Through a simple flowset composition language, ProgME can incorporate application requirements, adapt itself to circumvent the scalability challenges posed by the large number of flows, and achieve a better application-perceived accuracy. The modular design of ProgME enables it to exploit the surging popularity of multicore processors to cope with 7-Gb/s line rate. ProgME can analyze and adapt to traffic statistics in real time. Using sequential hypothesis test, ProgME can achieve fast and scalable heavy hitter identification.

III. PROPOSED METHODOLOGY

A data structure to store the time series of traffic so that the traffic information with the required granularity can be retrieved immediately. Our data structure is constructed of multiple trees. Each tree indicates one field of the traffic data (e.g., time, source IP address, or destination IP address). In each tree, the leaf nodes correspond to the traffic data of the finest-grained granularity, and a parent node corresponds to the coarser granularity that includes all of its children. Each node has the pointer to the root of the tree corresponding to the next field. Therefore, the traffic data with the required granularity can be obtained by continuing to search the nodes corresponding to the required granularity of the field.

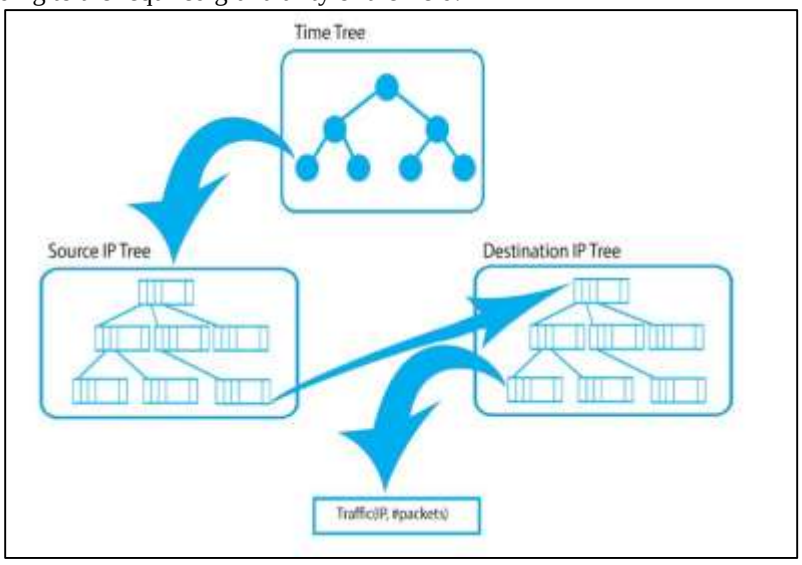

Fig. 1: Tree structure arrangement

A. Traffic Data:

Testing of the proposed data structure to store the time series traffic information is done using the log file from NS2. We included only a subset of the complete log file which contains the traffic information between 6 nodes in a time interval of one hour.

B. Initial Data Structure:

Our system uses a predefined time tree whose each node represents a range of time interval. Children of each node in time tree is the subset of the time interval of its parent node. ie; for example if a node store the time interval 1 to 30 minutes then its child nodes can have the values 1-10, 11-20 and 21-30 (time interval of children nodes can be changed as per our needs).
At the system initialization time we do not have any data in the data structures (source and destination tree), so the pointer to source tree from each time tree node will be set as null.

C. Populating Data Structure:

When a new time series information need to be inserted into the system, it will check for the time interval in which it actually belong (leaf node). Since we are aiming to attain all levels of granularity updating need to be done on all the nodes that it travel through to reach the leaf node. Each time node has a pointer to the root of a source tree. So data structure includes the same number of the source IP trees as the number of nodes in the time tree. For each node in source IP tree we have a pointer to destination IP tree. ie; same number of the destination IP trees as the number of nodes in the source IP tree. Destination IP tree nodes also contains the no of packets that are being transferred between that pair of source and destination pair. This arrangement of trees allows us to extract the required granularity of data based on the IP prefix. So for each entry in the log file we will populate the data structure in the above mentioned arrangement.

D. Time Tree:

The time tree is constructed so as to include the nodes corresponding to the time slots that are possible to be requested. Based on the defined possible time intervals, the structure of the time tree is determined. For example, we consider the case that the traffic information for one hour is stored and possible intervals can be 1 hour, 30 minutes, 10 minutes, 5 minutes.

E. Source IP Tree and Destination IP Tree:

The source and destination IP trees are constructed based on multiway trie. Parent node corresponds to the IP address prefix whose length is one digit shorter than its children. By implementing the IP trees using multiway trie we are able to reduce the level of IP trees to 12.

F. Working of the Proposed Approach:

The Graphical User Interface(GUI) consists of an option to insert source and destination IP address. We also provide the option to select the time slot between which the data transmission take place in the network. GUI is connected to function to traverse through the time tree based on the vale inserted in the GUI. When the corresponding node is found it will the function to traverse the source tree using the pointer to source tree from the time tree node. Second function traverse thought the source IP tree and find the corresponding node. This function is also enabled with searching of IP prefix and does not require a complete IP address all the time. When corresponding node in source IP tree is reached a new function is invoked to traverse the destination IP tree using the pointer to destination IP tree from the source IP tree node. This function traverse through the destination IP tree and find the corresponding node based on the input in the GUI. The node in destination IP tree contains the no of packets transmitted between that pair of source and destination in the given time interval. It is the returned to GUI to display as output.

G. Algorithm for Traversal in Time Tree:

1) Step 1: Start
2) Step 2: If root not empty go to step 2, Else End
3) Step3: If UTL and LTL equal to current UTL & Current LTL move to the Source Tree using the stored Reference, Else go to Step 4.
4) Step 4: If UTL/2 greater than Current UTL move to Right Child, Else go to Right Child
5) Step 5: Repeat Step 3 to 4
6) Step 6: Stop

H. Algorithm for Traversal in Source Tree:

1) Step 1: Start
2) Step 2: If root not empty go to Step 2, Else Return No Source Tree
3) Step 3: Extract next digit of the Source IP
4) Step 4: Move to child node with the index as the extracted IP digit
5) Step 5: Repeat the Step 3 to 4 till IP ends
6) Step 6: If Destination Tree reference not equal to Null move to that Destination Tree, Else Return No Source Tree
7) Step 7: Stop

I. Algorithm for Traversal in Destination Tree:

1) Step 1: Start
2) Step 2: If root not empty go to Step 2, Else Return No Source Tree
3) Step 3: Extract next digit of the Destination IP
4) Step 4: Move to child node with the index as the extracted IP digit
5) Step 5: Repeat the Step 3 to 4 till IP ends
6) Step 6: If current root not equal to null return Packet Count in that Node, Else Return No Destination Tree
7) Step 7: Stop

IV. EXPERIMENTAL RESULTS

The improvement made on the Source IP Tree and Destination IP Tree structure using the Multiway Trie reduced the insertion and retrieval time of the time series traffic data considerably. Comparing with the existing system which uses binary tree structure for implementation of IP Tree Structure, the improvement considerable reduced the number of levels in the IP Tree. In the existing system the number of levels in a fully filled IP Tree was 32 where as in the current system a fully filled IP Tree will have only a maximum of 12 levels. In the Multiway Trie based system the insertion of traffic data has to be made a maximum of 12 levels only likewise in case of search, search has to be conducted maximum of 12 levels.

V. CONCLUSION

By using the proposed data structure which enables the retrieval of the time series of network traffic with the requested granularity, we can reduce the overheads c for analyzing the network traffic data. Through reducing the level by using Multiway Tire structures we had reduced the insertion and retrieval time. Through experimental evaluation, we confirmed that our

proposed data structure provides immediate retrieval of the time series of the network traffic without large calculation overheads to store the network traffic data. Also a wide range of prefixed granularity is possible in our proposed system.

VI. FUTURE WORK

Our future work includes the discussion on more reduction of the nodes in the trees so as to save the memory size and the calculation time more. Also by using network traffic data in the data structure we can alter the routing table to avoid congestion in the network.

REFERENCES

[1] Yoshihiro Tsuji, Yuichi Ohsita, and Masayuki Muarata, "Data Structure Enabling Retrieval of Time Series of Traffic with the Requested Granularity" International Conference on Communication Systems (ICCS), 2014 IEEE , Nov. 2014
[2] T. Benson, A. Anand, A. Akella, and M. Zhang, "MicroTE: Fine Grained Traffic Engineering for Data Centers," in Proc. ACM CoNEXT, 2011, pp. 8:1–8:12.
[3] Tatsuya Otoshi, "Prediction-based control theoretic approach for robust traffic engineering," Master's thesis, Graduate School of Information Science and Technology, Osaka University, February 2014.
[4] T. Otoshi, Y. Ohsita, M. Murata, Y. Takahashi, K. Ishibashi, and K. Shiomoto, "Traffic Prediction for Dynamic Traffic Engineering Considering Traffic Variation," in Proc. IEEE Globecom, Dec. 2013, pp. 1592–1598.
[5] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," ACM Computing Surveys (CSUR), vol. 41, no. 3, p. 15, 2009. [5] A. Soule, K. Salamatian, and N. Taft, "Combining Filtering and Statistical Methods for Anomaly Detection," in Proc. ACM SIGCOMM IMC, Oct. 2005, pp. 31–31.
[6] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: Enabling Innovation in Campus Networks," ACM SIGCOMM CCR., vol. 38, no. 2, pp. 69– 74, 2008.
[7] P. Phaal, S. Panchen, and N. McKee, "InMon Corporations sFlow: A Method for Monitoring Traffic in Switched and Routed Networks," RFC 3176, 2001. [8] B. Claise, "Cisco Systems NetFlow Services Export Version 9," RFC 3954, Oct. 2004.
[8] L. Yuan, C.-N. Chuah, and P. Mohapatra, "ProgME: Towards Programmable Network Measurement," IEEE/ACM Trans. Netw., vol. 19, no. 1, pp. 115–128, Feb. 2011.
[9] D. E. Knuth, "The Art of Computer Programming. vol. 3: Sorting and Searching," Atmospheric Chemistry & Physics, vol. 1, 1973.
[10] W. Doeringer, G. Karjoth, and M. Nassehi, "Routing on Longest Matching Prefixes," IEEE/ACM Trans. Netw., vol. 4, no. 1, pp. 86–97, 1996.