IOSR Journal of Engineering (IOSRJEN) **www.iosrjen.org**
ISSN (e): 2250-3021, ISSN (p): 2278-8719
Volume 10, PP 57-60

Efficient Use of SDN in LTE for Better Cellular Traffic Control

Prof. Nilesh R. Gode[1], Dr. Bhavin Shah[2], Prof. Mohan Kumar[3]
[1, 2,3] (Electronics & Telecommunication Engineering Department,
Atharva College of Engineering/ Mumbai University, India)

Abstract: *This paper give detail study of the relationship between Software-Defined Networking (SDN) and Autonomic Network Management (ANM) under the prism of Long Term Evolution (LTE) Self-Organizing Networks (SONs). A complete revolution of mobile networks for accommodating over-growing demands of users, services and application. LTE Technologies also adopted much complex management requirements based on the softwarization of network resources. This will require a real time management based system on a hierarchy of complex decision making techniques that analyses historical, temporal and frequency network data. In this paper we discuss to solve the problem of wasting of network resources, SDN is considered as one of the best approaches. SDN traffic engineering gives the immense solution to enhance the network performance at both traffic and resource levels. Being the centralized authority SDN is able to give better results in network load balancing and improve delay and loss performance. We are using Load Balancing in SDN using Effective Traffic Engineering Method (LBTEM) in order to maintain the throughput of the network without any delay in packet reception with two layer control architecture.*

Keywords - *software defined; SDN ;Long Term Evolution LTE; Autonomic Network Management (ANM)*

I. INTRODUCTION

Various mechanisms are used to optimize the network performance dynamically by regulating and predicting the behavior of transmitting data. The traffic engineering mechanism is very well known among all. It is used in current data networks such as ATM and IP/ MPLS networks. For these types of networks the solutions are unfavorable for the next generation of cellular network and their network management due to two key reasons. First, todays networking applications require the underlying network architecture to be suitable for large amount of traffic. It should give discriminate services for different applications in a very short time period. Second, it should give better resource utilization in order to improve system performance with the increasing demand in cloud computing and data center [1]. Software defined networking (SDN) decouples the data plane and control plane, and assigns all routing decisions to an external device known as the controller. Being the central authority, the controller supervises all the routing decisions. It is fruitful with the help of Traffic engineering mechanisms and development of some new TE tools. Traffic engineering helps to analyze and measure the network traffic for the enhancement of the performance of network at both traffic and resource level [2] [3].

II. Traffic Engineering In Sdn

Traffic engineering means that the network traffic is measured and analyzed in order to enhance the performance of an operational network at both traffic and resource levels. The objective of Traffic Engineering is to solve congestion control problem to meet the diverse service and performance requirements. In SDN traffic engineering is much more efficiently implemented as compared to other conventional TE approaches such as ATM-, IP-, and MPLS-based TE approaches [4]. In SDN TE mechanism is much more successful and has many advantages as compared to other approaches. First, it gives global visualization by controlling the network centrally. Second, it gives the effective way of handling each individual infrastructure elements with the help of programmable network. Third is the data plane elements have a unified interface open to the controller for collecting the status and data plane programs. The fourth one is flow management is more flexible and efficient using multiple flow tables. TE technology gives focus on four areas, mainly flow management, fault tolerance, topology update and traffic analysis. In this paper, we give emphasis on the flow management in the SDN. The flow rule is set by the controller for the first packet of each flow. After the path is set up, all the subsequent packets in the same flow or different flow of same matching trait forward in the data plane path and do not need any control plane action. The controller updates the corresponding entry in the flow tables. If the combined traffic consists of the high number of new flows, a significant overhead has deferred at both the control and data plane. The latency may be increased due to the new forwarding rule. So the traffic engineering mechanism addresses the tradeoff between latency and load-balance [5].

Cognitive network management has been proposed as the current solution for this problem, in which the use of machine learning to develop self-aware, self-configuring, self-optimization, self-healing and self-protecting systems will enable cognitive network management. This technology is needed for managing a demanding infrastructure but one that yet has to present scalability and flexibility, such as that needed in LTE and higher cellular technologies.

III. Related Work

In this paper, our main focus is to consider the load balance approach in order to transfer the load of the network uniformly in the switches with low latency. With the increase number of new flows in the network, the congestion is increased at both the control and data plane. The solution to minimize this bottleneck is given by switch load balancing, controller load balancing and using multiple flow tables [6]. HyperFlow [7] is distributed control plane, which addresses the scalability issues with the help of network centralization. The decision making system is localized for minimizing the control plane response time to data plane requests. The synchronization system is used among the controllers to share the consistent network wide view and serve the request locally without the help of any remote node. All the controllers communicate among themselves using cross controller mechanism. Each switch is connected to the best controller in its neighborhood. Difane [8] is based on the concept of authority switches. The controller distributes the rules to authority switches. These switches handle all the packets in the data plane by using TCAM (Ternary Content Addressable Memory). The capacity of TCAM is limited, and it is very much power hungry. Upon receiving the packet if it does not match with any of the cached rules, then the switch directs it to the authority switch. All authority rules are stored in the authority switches. The authority switch sets the cached rules of the network. For this Difane does not need any modification in data plane operation, it simply needs a minor software operation, modification in the control plane of the authority switches. Onix[9] is based on cluster of one or more physical servers working on a distributed platform. The network control logic is implemented in Onix API. The state information of any in-network elements can be tracked by a data structure called NIB (Network Information Base) maintained by Onix. It is a graph of all networking entities within a network topology. This information is shared among multiple controllers to solve scalability problem. BalanceFlow[10] is based on the distributed controller concept to balance the load of the controller. It is useful for balancing the load of an overloaded controller by dynamically distributing the flow-requests among controllers. The load is balanced among the other low-loaded controllers. All controllers maintain the flow-requests information and share among other controllers in the network using cross controller communication. There are two types of controller present in the network. One is the super controller and more than one normal controller. The super controller is responsible for distributing the messages in order to balance the load of all controllers. The controller for which the average number of flow requests is more than a threshold value is considered as imbalance controller. This load is adjusted by the super controller with the help of super controller performance and network elements. Kandoo[11] is an example of hierarchical controller. It creates two level hierarchy. One is local and the other is non local. The former executes the local applications. The later type runs non local control applications. This is known as root controller and is logically centralized. The root controller controls all the local controllers, which in turns control the switches connected to the local one. The local controller controls all the switches by installing the flow entries as directed by the root controller.

IV. Proposed APPROACH

For efficient routing it is necessary to free the router from the computation overburden. In order to solve this problem SDN plays an essential role by centrally controlling the routing decision. At the same time delay is introduced in packet processing that creates another problem of increasing the overall load of the network. Perfect load balancing in the network at the same time minimizing delay are the two motivations of this paper. In [9] the real time traffic of the China Education Network is monitored and it is observed that the maximum number of traffic flows is 3 million per second. So if we make the flow table entry for the OFdevices as null, then it generates 3 million PACKET_IN messages. Also, after a certain point of time the PACKET_IN messages to the controller increases exponentially. For controller load balancing some of the of devices needs to be migrated to some other controller. For this the first job is to select the overloaded controller and the victim OF device. This selection procedure incurs an additional delay in packet processing. At the same time if in some situations the migrated OF device need to be reverted back to the original controller due to overburden in targeted controller, causing delay in the network. To minimize these various problems in the network, we make it a multi objective optimization problem. The solution to this problem is given by selecting an alternative switch for overburden switch. System Model In this section we propose the overall system model of LBTEM (Load Balancing in SDN using Effective Traffic Engineering Method). Assume the network is presented by G(N, E), where N (N1, N2, N3,… Ni) represents the number nodes consists of the of devices for network packet transmission and E (E1, E2, E3,… Ej) represents the edge connecting the of devices.

The orchestration of the network can be thought of as resolving a number of independent and in some cases interdependent management objectives across a number of key objectives such as:
1. Provisioning: ensure that the network is adequately provisioned with resources sufficient to deal with current demand levels while maintaining QoS at an agreed level.
2. Security Management: Protect network data and its performance through accurately detecting intrusion, privacy and denial of service as well as autonomous anomaly detection.
3. QoS support: Network Slicing supports several defined QoS levels simultaneously and kept logically isolated by the same physical network.
4. Fault Tolerance: The network should be able to recognize emerging faults or error conditions and pre-emptively deal with them, or intercept unexpected faults or errors as quickly as possible to minimize any reduction in QoS.
5. Energy Efficiency: Optimizing the source of energy, for example maximizing the use of renewable and energy efficiency may be a factor in the selection of computing resources to support network functions.
6. New components: The operator should also be able to add new objectives and to change their priorities as well.

V. Evaluation

Load Balancing in SDN using Effective Traffic Engineering Method (LBTEM) is evaluated using Mininet [14]. Mininet is used for deploying large network with the cost of limited resources. It is helpful to enhance the research direction both in SDN and OpenFlow. On a simple computer Mininet is very much useful to run the code without modifying it. The code runs interactively on a virtual hardware. An alternative solution to Mininet is very expensive, no doubt very fast and accurate. In some simulator is very cheap and comparatively slow also. By considering all these parameters Mininet performance is accurate and scalable [15]. We have studied the performance parameters such as throughput, energy efficiency and latency behavior. At the same time the proposed one is compared with DIFANE.

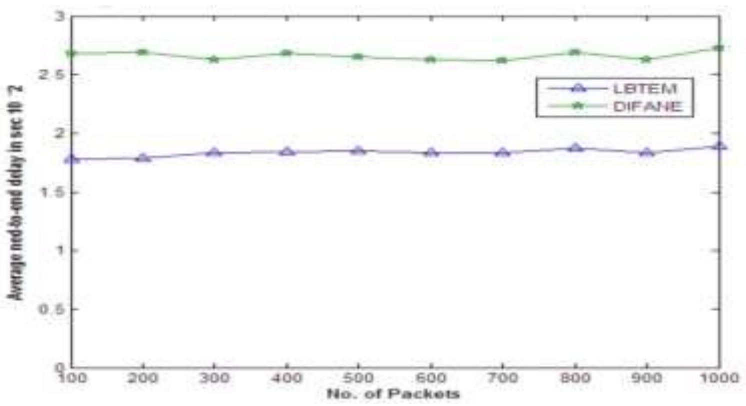

Fig 1. Energy consumption Vs No. of Packets

The plot for energy consumption vs. number of packets, is shown in Fig. 2. It is observed from the figure that the energy consumption in LBTEM is lower than DIFANE. This is because in LBTEM by using two types of controller the overall energy consumption of the network is minimized.

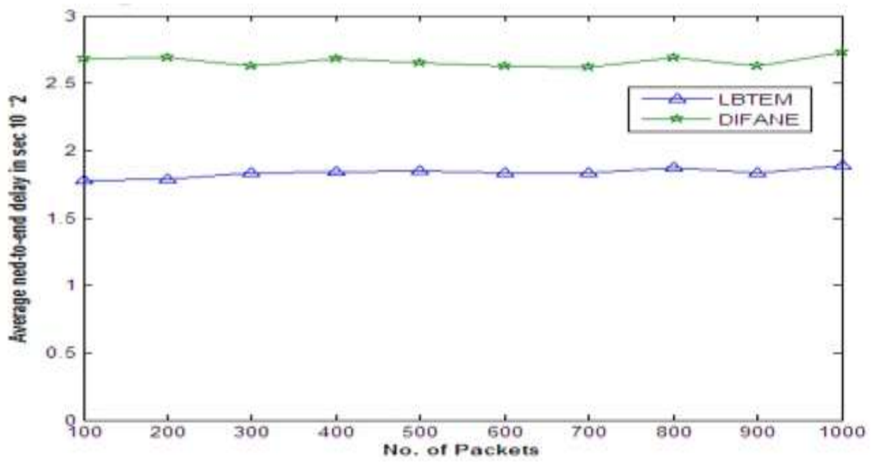

Fig 2. Average End to end delay Vs. No. of Packets

Next, we plot the graph for end-to-end delay vs. number of packets, is shown in Fig. 3. It is observed from the figure that the average end-to-end delay in LBTEM is lesser than that of DIFANE. Lower end-to-end delay in LBTEM is attributed to more balancing of traffic load.

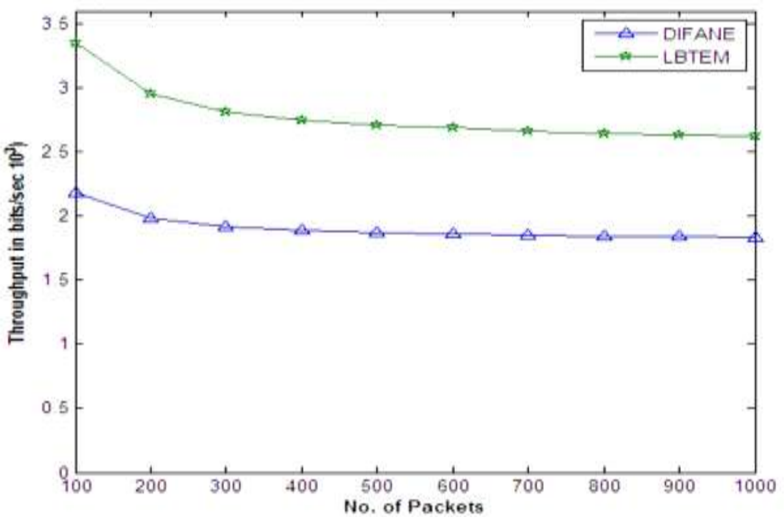

Fig 3. Throughput Vs. No. of Packets

VI. Conclusions

In Load Balancing in SDN using Effective Traffic Engineering Method (LBTEM), we balance the traffic load of the SDN by using two layer architecture. This architecture is possible with the help of one super controller and many numbers of sub controller. In switch the flow table is updated with the help of sub controller. The moment when the sub controller is overloaded the network is balanced with the help of super controller. We have simulated LBTEM with Difane using Mininet simulator. It is observed from the simulation result that the network load is not only balanced, but also at the same time the energy consumption is minimized without affecting the delay.

References

[1] Zhou, Yuanhao, et al. "A load balancing strategy of SDN controller based on distributed decision." Trust, Security and Privacy in Computing and Communications (TrustCom), 2014 IEEE 13th International Conference on. IEEE, 2014. Hasan, Syed Faraz. Emerging trends in communication networks. Springer, 2014

[2] Park, Sang Min, Seungbum Ju, and Jaiyong Lee. "Efficient routing for traffic offloading in Software-defined Network." Procedia Computer Science 34 (2014): 674-679.

[3] Akyildiz, Ian F., et al. "A roadmap for traffic engineering in SDN- OpenFlow networks." Computer Networks 71 (2014): 1-30.

[4] Tootoonchian, Amin, et al. "On Controller Performance in Software- Defined Networks." Hot-ICE 12 (2012): 1-6.

[5] Awduche, Daniel, et al. Overview and principles of Internet traffic engineering. No. RFC 3272. 2002. Tootoonchian, Amin, and Yashar Ganjali. "Hyperflow: A distributed control plane for openflow." Proceedings of the 2010 internet network management conference on Research on enterprise networking. 2010. [8] Yu, Minlan, et al. "Scalable flow-based networking with DIFANE." ACM SIGCOMM Computer Communication Review 40.4 (2010): 351-362.

[6] Yonghong, Fu, et al. "A dormant multi-controller model for software defined networking." China Communications 11.3 (2014): 45-55.

[7] Hu, Yannan, et al. "Balanceflow: controller load balancing for openflow networks." Cloud Computing and Intelligent Systems (CCIS), 2012 IEEE 2nd International Conference on. Vol. 2. IEEE, 2012.
[8] Hassas Yeganeh, Soheil, and Yashar Ganjali. "Kandoo: a framework for efficient and scalable offloading of control applications." Proceedings of the first workshop on Hot topics in software defined networks. ACM, 2012. [12] Kim, Hyojoon, and Nick Feamster. "Improving network management with software defined networking." IEEE Communications Magazine 51.2 (2013): 114-119.
[9] Giroire, Frédéric, Joanna Moulierac, and Truong Khoa Phan. "Optimizing rule placement in software-defined networks for energy-aware routing." Global Communications Conference
[10] Karamjeet Kaur1, Japinder Singh2 and Navtej Singh Ghumman, “Mininet as Software Defined Networking Testing Platform”, International Conference on Communication, Computing & Systems (ICCCS–2014).
[11] Mininet. http://mininet.org/[M].