A Load Balancing Method Based on SDN

Mao Qilin, Shen WeiKang

Nanjing Institute of Technology, Nanjing, Jiangsu, 211167, China
maoql20011@163.com

***Abstract*—Server load balancing based on SDN compared with the traditional load balancing method, can effectively improve the performance of the server load balancing, and reduce the complexity of implementation. Openflow is south interface protocol of SDN switch. The core of server load balancing based on SDN is dynamic building Openflow flow tables. This paper puts forward a dynamic flow table design algorithm which based on "single flow table" and "group flow table" combination, "Single flow table" can accurately monitor traffic of each client, "Group flow table" can effectively category client hosts. This algorithm does not only effectively avoid excessive number of flow tables, and better solve the defect that flow table matching range is too wide, experiments show that the algorithm has good feasibility and higher traffic scheduling performance of network.**

Keywords- Load Balancing; SDN, OpenFlow; Group Flow Table

I. INTRODUCTION

In recent years, with the rapid expansion of Internet business, using the load balancing technology to deal with this challenge has become a backend server necessary measure. The load balancer is a bridge between the network and server, load balancer usually need to learn the health status of the server, also don't need to be able to set up the network protocol, packet content information to modify or read. Traditional load balancing device are considered in the design of the operation condition of the server computer equipment (such as CPU utilization) to make a decision, but considering the network traffic is less load of network equipment, the lack of fine grained monitoring and scheduling, the main reason is that the current network system is a relatively closed system. Although the traditional router can allocate bandwidth between different paths, but only a thick line of control. In general, professional load balancing server needs to be done from two to seven layers of data processing, so it is easy to become the bottleneck of the whole service system.

With the emergence of Software Defined Networking (SDN) technology, network openness was the largest. Proposing the idea of separating and transmitting, SDN can obtain the global view of the network. Thus far, Openflow protocol which is the most widely used in a SDN south interface protocol is the concrete embodiment of the concept of SDN. Openflow offers us another kind of design method of load balancing, the server load balance based on SDN compared with the traditional load balancing method. It has a simple implementation and high performance characteristics. In this paper is proposed based on a "single flow table" and "group flow table" combination of dynamic flow table design algorithm. "Single flow table" focuses on traffic surveillance; "group flow table" mainly completes traffic forwarding function.

II. THE MAIN CHARACTERISTIC OF SDN

Software defined networking is a new kind of network architecture, the design concept is to network the decouple control plane and data forwarding plane, and programmable network oriented business layer. The main characteristic of SDN is shown as the following several aspects:

1) centralized control

SDN control plane and forwarding plane separation, forwarding device can focus forward and their function more simple, centralized control plane has better flexibility and innovation. The centralization of the control can simplify the network operation management, improve the speed of business configuration, and is beneficial to realize the rapid upgrade and innovation of the network. Centralized control is beneficial to global optimization of network, such as traffic engineering and load balancing can be considered from the perspective of whole business.

2) open the software programming interface

Through an open API, customized network business can be provided to customers. The upper application can be seamless docking, and the network application can change the network to make it meet the needs of their business.

3) network virtualization

Logical network is separated from the physical network to ensure that the logical network is not limited by the physical network. Logical network can be any combination, mobile. Network virtualization can easily make the actual physical network into multiple logical virtual networks to meet the needs of different business and network innovations.

III. THE BASIC MODEL OF LOAD BALANCING BASED ON SDN

Traditional server load balancing network model as shown in Figure.1 , the client connects load balancing server through the virtual IP (VIP) , load balancing server according to the health of the backend server select the corresponding load balancing algorithm to redistribute the access of external client to different backend server. Load balancing server must have the ability to keep the session, namely all the packets with the same TCP connection must be forwarded to the same backend servers. Traditional load balancer must have the function of network address translation (NAT), which can be realized by modifying the TCP packet source IP, source port, destination IP port as well as the destination

978-1-4673-7143-8/15 $31.00 © 2015 IEEE
DOI 10.1109/ICMTMA.2015.13

MAC address and so on, thus the traditional load balancer is usually treated as layer 2/3 switch equipment.

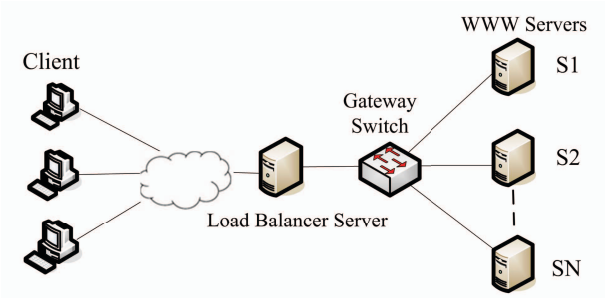

Figure 1. The traditional server load balance network model

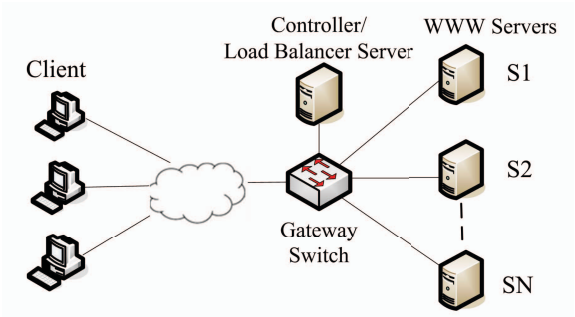

Figure 2. The server load balance network model based on SDN

The server load balancing network model based on SDN is shown in Figure.2. In this model the server load balancing no longer directly modified TCP packet source IP, source port, destination IP port nor the destination MAC address, but through the way of distributed flow table by SDN switches to complete the NAT function. SDN load balancer only used to generate, modify, or delete rules of flow table, no longer to forward specific client any packets. SDN load balancer flow table is mainly based on health inspection and the corresponding load balancing algorithm.

IV. OPENFLOW FLOW TABLE MATCHING RULES AND ALGORITHM DESIGN

In OpenFlow protocol, the controller influences the transmission through the SDN switches that are installed on the flow table, so the design flow table of matching rules and the corresponding action is the core of the server load balance based on SDN. Setting the rules of load balance for flow table each server's health has to be considered and also the TCP packets session remains and the flow of SDN switches have to be considered.

If different flow table is set for each client and lead to too many flow table, it may affect the processing performance of server load balancing. For example, using source IP address for matching rules of flow table is simple to implement, but easy to produce a large number of flow tables. Obviously this way is not suitable for the load to balance the flow table setting way. Different client processing must be classified somehow. This can be done through the flow table of the wildcard, OpenFlow protocol of IP address used is similar to the CIDR (Classless Inter-Domain Routing) the IP address of the clustering methods, such as low 10.0.0.100/31, namely one bit is a wildcard, such 10.0.0.100 and 10.0.0.101 can use the same flow table matching rules.

Paper [1] is proposed to minimize the flow table rule set of binary number matching method, but the premise of this paper is that the client's IP address distribution is uniform. This method does not take into account the actual traffic situation. This flow table design concept is based on the actual access client source IP address distribution and the actual flow in the rules of flow table. At the same time the rules' generation, modification and destruction shall ensure the TCP session.

In order to illustrate the purpose of this algorithm, a single client flow table rule is known as the "Single Flow table"(SF), and multiple client flow table rules are referred to as the "Group Flow table" (GF). All SF priorities are higher than GF priorities. The characteristics of GF are shown in Figure.3, each GF covered by IP address range cannot overlap, and multiple GF may be used to forward packets to the same server. GF can be modified to override the host IP address range, in order to dynamically adjust to different server traffic; GF can be split or merged. SF life cycle is very short, but GF has a long life cycle.

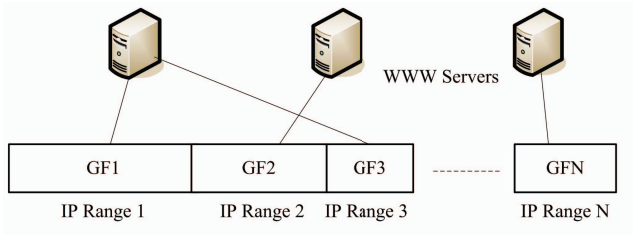

Figure 3. Groups of flow table configuration diagram

In this paper, the basic idea of the flow table rule algorithm is as follows:

1) SF set is initialized to null, establishment GF set of rules covering all IP addresses, establishment of the status description words of the client. States are divided into three types: built-in single-stream, stream table processing, workflow forms processing.

2) When a client is accessing the VIP, first check the status of the client. If the status is:

"to build the single flow table" , go to step 3.

"single flow table processing" , go to step 4.

"group flow table processing", go to step 5.

3) When the status is "to build the single flow table", the client is considered to be a "new client". The client will create a SF, put it into the single flow table rule set, and client status changes to "single flow table processing". At the same time, real-time monitoring of the single flow table processing packet number and traffic takes place.

4) If no new packets are processed within a certain period of time and the single flow table can be deleted, go to step 6, otherwise go to step 5.

5) Through the real-time monitoring of each GF processing packet number and traffic, the corresponding set

of GF for SF is searched based on the health of the backend server and the traffic status of single flow. Then is checked whether the GF matching rules need to be modified. If the rules have to be modified, depending on the source address matching algorithm the GFT and adjacent GF matching rules will be modified the SF will be removed, and the client status will be changed to "group flow table processing". The removal of the SFT should guarantee the end of the TCP session, SF should be deleted after the load balancing algorithm module has received the TCP FIN packet.

6) server load balancing offers, regularly or according to some emergency such as if the server goes down, real-time dynamic maintenance of GF, including GF IP address range adjustment, adding or removal of GF. When the GF changes have taken place, the IP address has been adjusted, the client status will be set to "to build the single flow table" state.

Why SF should be designed in the algorithm above? The aim is to avoid GF's frequent changes to the design, but also in order to understand the actual traffic better, thus predict the traffic produced by the host in the future. GF's change is usually based on monitoring the flow changes and the server's health. For example if GF's flow has increased significantly, then during a certain time period the packets that are sent to the backend server have larger changes. At this moment, some host settings for SF can be restored in order to find client hosts with larger traffic. Necessary information for the subsequent changes in the GF should be provided.

V. THE REALIZATION AND ANALYSIS OF THE SYSTEM

Based on a small SDN network experimental environment the above flow table matching rules and algorithm can be implemented, it's model is shown in Figure.4. Open vSwitch software is installed on the SDN switch. Controllers are using open source OpenDayLight (ODL) software, the programming interface is ODL REST API. More PC hosts are adopted to simulate the external client in order to access WWW servers. During initialization time the client IP addresses are divided into three groups: 10.0.0.8 ~ 10.0.0.11 (GF1), 10.0.0.12 ~ 10.0.0.15 (GF2), 10.0.0.16 ~ 10.0.0.23 (GF3). Using the source IP address to match the design flow table rules, three groups of the flow table matching rules are:10.0.0.8/30, 10.0.0.12/30, 10.0.0.16/29. GFT1, GFT2 and GFT3 will forward the traffic respectively to the servers S1, S2, and S3. Assuming the three servers all have the same weight of traffic handling capacity and the all clients access server VIP address is 10.0.0.118, during the system running, if server S2 has overload, and server S1 has light load, then the IP address matching rules of the GF2 can be changed to 10.0.0.14/31. At the same time, the GF1 matching rules can be changed to 10.0.0.8/29, so it will host IP addresses 10.0.0.12 and 10.0.0.13. The traffic on those IP addresses is forwarded to the server S1 to reduce the load on server S2.

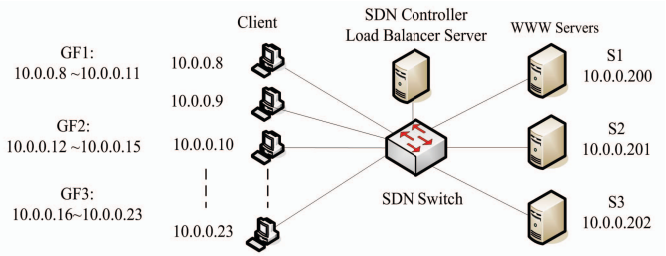

Figure 4. Structural model experiment system network configuration

Specific flow table is set up as shown in Table 1, take GF3 (forwarding data to the S1) as an example:

A. *The flow table of the installation of the entrance to the packet*

Matching conditions: the source IP address matching 10.0.0.16/29, priority: 400, action: modify the destination IP address to 10.0.0.200, modify the destination MAC address to MAC address of host (10.0.0.200), and specify the corresponding switches physical out port (OF|3).

B. *The flow table of the installation of the response packet*

Matching conditions: the source IP address matching 10.0.0.200, priority: 500, action: modify the source IP address to 10.0.0.118, modify the MAC address to MAC address of host (10.0.0.118), and specify the corresponding switches physical out port (OF|5).

Load balancing system based on SDN is divided into three modules: server performance detection module (SPDM), network performance detection module (NPDM) and flow table management module (FTMM). SPDM checks the health of WWW server. The main checking indicators are CPU and memory utilization, through the server hosting program the performance indicators will be periodically reported to the load balanced SPDM processing module. NPDM is responsible for monitoring the statistics of each flow table in SDN switches and managing the flow table together with the FTMM. FTMM is the core module of the system; the main functions are to maintain and manage the flow table and assess the processing performance of the flow table to the load balancer.

The system test is initiated when the client requests the HTTP, then the traffic statistics of each flow table are read through the REST API function provided by the ODL, including the number of packets, the number of bytes, and the time interval. Experimental data show that the traffic of the packet received by the server remains balanced. Then the frequency of a certain group of clients is adjusted for accessing the backend server in a very short period of time one server's traffic had a significant increase. Once the rapid increase was detected, the load balancing algorithm would quickly adjust the coverage of the flow table. The relevant traffic is transmitted to a lightly loaded server to achieve the purpose of server load balancing.

Table 1: design rules of ODL flow table and traffic statistics. 10.0.0.16 is showed in the ODL software interface, the actual should be 10.0.0.16/29

NW Src	Actions	Packet Count	Duration Seconds	Priority
10.0.0.16	SET_NW_DST=10.0.0.200 SET_DL_DST=74:27:ea:23:bb:47 OUTPUT=OF|3	696070	3803	400
10.0.0.200	SET_NW_SRC=10.0.0.118 SET_DL_SRC=74:27:ea:23:88:8f OUTPUT=OF|5	2782902	3806	500

VI. CONCLUSIONS

The key of server load balancing based on SDN is setting the SDN flow table. On the one hand, the number of flow table cannot be too large, otherwise the performance of the dynamic maintenance of flow table is affected, and so fuzzy matching of the source IP address is usually used. On the other hand, the range of matching IP of the flow table cannot be overly broad; this would be bad for precise dispatching of network traffic. In this paper, the setting algorithm of the flow table is to some extent a trade-off of the above requirements. It put forward the "single flow table" and "group flow table" design algorithm. The "single flow table" life cycle is very short, but "group flow table" has a long one. The "group flow table" can effectively avoid the problem of excessive number of flow table, while the "single flow table" again to monitor traffic of each client and provide the basis for changes of the "group flow table ".

ACKNOWLEDGMENT

Project: The key research of SDN business platform for the smart grid (BY2013095-4-07)

The project by the future network prospective research project funding, that provide by Jiangsu Future Networks Innovation Institute (China).

REFERENCES

[1] Richard Wang, Dana Butnariu, and Jennifer Rexford. OpenFlow-Based Server Load Balancing Gone Wild. Hot ICE, 2011.

[2] N. Handigol, S. Seetharaman, M. Flajslik, N. McKeown, and R. Johari. Plug-n-Serve: Load-balancing web traffic using OpenFlow, Aug. 2009. Demo at ACM SIGCOMM.

[3] M. Koerner, O. Kao. Multiple service load balancing with OpenFlow. IEEE HPSR, 2012.

[4] Li Chuan Chen, Hyeon Ah Choi. Approximation algorithms for data distribution with load balancing of Web Servers[A]. In Proceedings of IEEE International Conference on Cluster Computing[C], 2001:274-281

[5] 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 Computer Communications Review, 38(2):69–74, 2008.

[6] ONF white paper. Software-Defined Networking: The New Norm for Networks. https://www.opennetworking.org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf

[7] Dutcher, Bill. The NAT Handbook: Implementing and Managing Network Address Translation. New York: John Wiley and Sons, 2001.

[8] Adam Cahn, Juan Hoyos, Matthew Hulse, Eric Keller. University of Colorado,Boulder. Software-Defined Energy Communication Networks: From Substation Automation to Funture Smart Grids. in IEEE Conf. on Smart Grid Communications (SmartGridComm). To appear Oct., 2013.

[9] Tony Bourke. Server Load Balancing. O'Reilly Media; 1 edition, 2001.

[10] Opendaylight website. http://www.opendaylight.org

[11] OpenFlow Switch Specification, version 1.0, 1.3, 1.4 .

[12] Soheil Hassas Yeganeh, Yashar Ganjali. University of Toronto. Kandoo: A Framework for Efficient and Scalable Offloading of Control Applications. ACM SIGCOMM HotSDN, 2012.

[13] Adam Cahn, Juan Hoyos, Matthew Hulse, Eric Keller. University of Colorado,Boulder. Software-Defined Energy Communication Networks: From Substation Automation to Funture Smart Grids. in IEEE Conf. on Smart Grid Communications (SmartGridComm). To appear Oct., 2013.

[14] S.Kandula, S.Sengupta, A.Greenberg, P.Patel and R.Chaiken. The Nature of Data Center Traffic: Measurements & Analysis. ACM IMC 2009.