Computer Networks 68 (2014) 95–109

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Improving the performance of load balancing in software-defined networks through load variance-based synchronization

Zehua Guo [a,*], Mu Su [b], Yang Xu [b], Zhemin Duan [a], Luo Wang [b], Shufeng Hui [c], H. Jonathan Chao [b]

[a] School of Electronics and Information, Northwestern Polytechnical University, Xi'an 710072, China
[b] Department of Electrical and Computer Engineering, New York University Polytechnic School of Engineering, NY 11201, USA
[c] Department of Computer Science and Engineering, New York University Polytechnic School of Engineering, NY 11201, USA

ARTICLE INFO

Article history:
Received 30 May 2013
Received in revised form 7 November 2013
Accepted 4 December 2013
Available online 26 February 2014

Keywords:
Software-Defined Networking
Load balancing
Multiple controllers
Controller state synchronization

ABSTRACT

Software-Defined Networking (SDN) is a new network technology that decouples the control plane logic from the data plane and uses a programmable software controller to manage network operation and the state of network components. In an SDN network, a logically centralized controller uses a global network view to conduct management and operation of the network. The centralized control of the SDN network presents a tremendous opportunity for network operators to refactor the control plane and to improve the performance of applications. For the application of load balancing, the logically centralized controller conducts Real-time Least loaded Server selection (RLS) for multiple domains, where new flows pass by for the first time. The function of RLS is to enable the new flows to be forwarded to the least loaded server in the entire network. However, in a large-scale SDN network, the logically centralized controller usually consists of multiple distributed controllers. Existing multiple controller state synchronization schemes are based on Periodic Synchronization (PS), which can cause undesirable situations. For example, frequent synchronizations may result in high synchronization overhead of controllers. State desynchronization among controllers during the interval between two consecutive synchronizations could lead to forwarding loops and black holes. In this paper, we propose a new type of controller state synchronization scheme, Load Variance-based Synchronization (LVS), to improve the load-balancing performance in the multi-controller multi-domain SDN network. Compared with PS-based schemes, LVS-based schemes conduct effective state synchronizations among controllers only when the load of a specific server or domain exceeds a certain threshold, which significantly reduces the synchronization overhead of controllers. The results of simulations show that LVS achieves loop-free forwarding and good load-balancing performance with much less synchronization overhead, as compared with existing schemes.

1. Introduction

Software-Defined Networking (SDN) is a promising networking technology that provides network operators more control of the network infrastructure. SDN technology decouples the control plane logic from the data plane by

* Corresponding author. Tel.: +1 6467094105.
E-mail addresses: guolizihao@hotmail.com (Z. Guo), msu01@students.poly.edu (M. Su), yangxu@poly.edu (Y. Xu), zhemind@nwpu.edu.cn (Z. Duan), lwang12@students.poly.edu (L. Wang), shufeng.hui@nyu.edu (S. Hui), chao@poly.edu (H.J. Chao).

http://dx.doi.org/10.1016/j.comnet.2013.12.004

moving networking control functions from forwarding devices (e.g., switches, routers) to a logically centralized controller so that the networking functions can be implemented by software. One enabler of SDN technology is OpenFlow [1], which provides global visibility of flows in the network by enabling flow-level control over Ethernet switching. Currently, many companies and organizations are working on deploying OpenFlow on backbone and campus networks, such as Internet2 [2], Global Environment for Network Innovation (GENI) [3], and Google's WAN-B4 [4].

Typically, an SDN network relies on a logically centralized controller using the global network knowledge to operate the network. In the SDN network, the controller conducts packet processing by matching forwarding rules, which can be installed in the flow tables of switches either reactively (e.g., when a new flow arrives) or proactively (e.g., controller installs rules in advance). When a new flow traverses a domain for the first time, the controller selects a forwarding path for the new flow based on the current global network status and configures the forwarding path by reactively installing rules on related switches. The reactive operation using the current global network status makes the improvement of applications possible. For the application of load balancing, when a new flow enters a domain for the first time, the controller conducts Real-time Least loaded Server selection (RLS) for this domain to select the current least loaded server as the destination server of the new flow based on the current network status. After a series of RLSs for related domains, the new flow's forwarding path to the least loaded server is determined, and the packets of this flow are forwarded via this path. The detail of RLS are elaborated in Section 2.

However, having one centralized controller creates problems including poor scalability, responsiveness, and reliability, which have significantly hindered and detained the deployment of SDN technology in large-scale production and data center networks [5,6]. To overcome the above problems, a simple solution is to use multiple distributed controllers working together to achieve the function of the logically centralized controller (e.g., HyperFlow [5] and ASIC [6]), which can benefit from the scalability and reliability of the distributed architecture while preserving the simplicity of the centralized system. In a multi-controller multi-domain SDN network, each controller runs one domain simultaneously, and it only handles the local area switches in its domain. Currently, most of the controllers applied in the SDN network can be categorized as *Local State-aware Controllers (LSCs)*. An LSC could, in real-time, acquire the state of servers in its controlled domain by pulling statistics from edge switches of its domain. However, the LSC obtains the state of servers in domains managed by other controllers through controller state synchronization. The LSC stores the state of servers in its Network Information Base (NIB) and uses its NIB to make forwarding decisions for new flows [7]. A NIB may contain different contents based on the need of network applications [7–9]. In this paper, we focus on the flow-based, web load balancing, where the NIB of each controller contains the load of each server (e.g., the utilization of the link connecting with each server [7]).

One of the major challenges when using multiple controllers in the SDN network is to synchronize the state among controllers. The state desynchronization among controllers significantly degrades the performance of the application. Levin et al. [7] study the controller state synchronization problem for the application of load balancing in the SDN network and propose two controller state synchronization schemes based on Periodic Synchronization (PS): Link Balancer Controller (LBC) and Separate State Link Balancer Controller (SSLBC). Both of these schemes initiate state synchronization among controllers within a fixed period. Using LBC, each controller directs newly arrived flows to the current least loaded server in the entire network based on its NIB, which is updated and synchronized periodically by controller state synchronizations. However, good load-balancing performance using LBC requests frequent synchronizations, which overly burden controllers. To achieve good performance with less synchronization overhead of controllers,[1] SSLBC is further developed. In SSLBC, each controller divides its NIB into two parts: the local NIB containing the load of servers in the local domain and the remote NIB containing the load of servers in domains except the local domain. Using SSLBC, the remote NIB is updated periodically by controller state synchronizations, while the local NIB is updated in real time. The local domain's controller is aware of the real-time load variation of servers in the local domain by the real-time updated local NIB, which enables the controller to select the least loaded server based on accurate local domain status. Thus, SSLBC achieves better load-balancing performance with the same number of synchronizations as LBC. However, as will be detailed in Section 3, SSLBC may lead to several undesired exceptional situations (e.g., forwarding loops [7,10–12] and black holes [7,10,11]) due to state desynchronization among controllers.

In this paper, we investigate the application of load balancing in the SDN network with multiple domains managed by multiple controllers. We propose a new type of controller state synchronization scheme, Load Variance-based Synchronization (LVS), to improve load-balancing performance by eliminating forwarding loops and to lower synchronization frequency. We first introduce load balancing in the SDN network in detail and differentiate it from load balancing in the IP network. Based on the description, we analyze two problems in existing PS-based schemes: the high controller synchronization overhead problem caused by frequent synchronizations and the forwarding-loop problem caused by state desynchronization among controllers during the interval between two consecutive synchronizations. Then, we solve the two problems with two specific LVS-based schemes: Least loaded Server Variation Synchronization (LSVS) and Least loaded Domain Variation Synchronization (LDVS). Compared with PS-based schemes, LVS-based schemes achieve low synchronization frequency by conducting effective controller state synchronizations only when the load of a specific server or domain exceeds a certain threshold. The results of simulations

[1] We use the terms synchronization overhead of controllers and the number of synchronizations interchangeably in this paper.