CDLB: A Cross-Domain Load Balancing Mechanism for Software Defined Networks in Cloud Data Center

CDLB: a cross-domain load balancing mechanism for software defined networks in cloud data centre

Weiyang Wang

School of Electronic Information and Electrical Engineering,
Shanghai Jiao Tong University,
Shanghai, China
Email: wywangsh@163.com

Mianxiong Dong* and Kaoru Ota

Department of Information and Electric Engineering,
Muroran Institute of Technology,
Muroran, Japan
Email: mx.dong@csse.muroran-it.ac.jp
Email: ota@csse.muroran-it.ac.jp
*Corresponding author

Jun Wu, Jianhua Li and Gaolei Li

School of Electronic Information and Electrical Engineering,
Shanghai Jiao Tong University,
Shanghai, China
Email: junwuhn@sjtu.edu.cn
Email: lijh888@sjtu.edu.cn
Email: gaolei_li@sjtu.edu.cn

Abstract: Currently, cross-domain load balancing is one of the core issues for software defined networks (SDN) in cloud data centre, which can optimise resource allocation. In this paper, we propose a cross-domain load balancing mechanism, CDLB, based on Extensive Messaging and Presence Protocol (XMPP) for SDN in cloud data centre. Different from poll method, XMPP based push model is introduced in the proposed scheme, which can avoid wasting network and computing resources in large-scale distributed network environment. The proposed scheme enables all the controllers in the flat distributed control plane to share the same consistent global-view network information in real time through XMPP and XMPP publish/subscribe extension. Thus, the problem of non-real time information synchronisation can be resolved and cross-domain load balancing can be realised. The simulations show the efficiency of the proposed scheme.

Keywords: cloud data centre; extensive messaging and presence protocol; XMPP; push model.

Reference to this paper should be made as follows: Wang, W., Dong, M., Ota, K., Wu, J., Li, J. and Li, G. (2019) 'CDLB: a cross-domain load balancing mechanism for software defined networks in cloud data centre', *Int. J. Computational Science and Engineering*, Vol. 18, No. 1, pp.44–53.

Biographical notes: Weiyang Wang received his BS in Electronic Information Engineering from Xidian University, Shanxi, China in 2013 and is currently pursuing his MS in Information and Communication Engineering in Shanghai Jiao Tong University, Shanghai, China. He participates in many national projects, such as National Natural Science Foundation of China, National '973' Planning of the Ministry of Science and Technology, China, etc. His research interests include software defined network, cloud computing, etc.

Mianxiong Dong received his PhD from the University of Aizu, Japan. Currently, he is an Assistant Professor at the Department of Information and Electronic Engineering, Muroran Institute of Technology, Japan. His research interests include wireless sensor networks, software-defined networks, etc. He has published more than 130 research papers in international journals, conferences and books. He serves an Associate Editor of *IEEE Communications Surveys & Tutorials*, *IEEE Access*, etc. and Guest Editor of *ACM Transactions on Multimedia Computing, Communications and Applications (TOMM)*, *IEEE Transactions on Computational Social Systems (TCSS)*, etc. And also, he serves as the Symposium Chair of IEEE GLOBECOM 2016.

Kaoru Ota received his PhD degrees in Computer Science and Engineering from the University of Aizu, Japan in 2012. She is currently an Assistant Professor at the Department of Information and Electronic Engineering, Muroran Institute of Technology, Japan. Her research interests include wireless networks, cloud computing, and cyber-physical systems. She is the Editor of *IEEE Transactions on Vehicular Technology* (TVT), *IEEE Communications Letters*, etc, as well as a guest editor of many *ACM/IEEE Trans*. She was a Research Scientist at the A3 foresight program (2011–2016) funded by Japan Society for the Promotion of Sciences (JSPS), NSFC of China, and NRF of Korea.

Jun Wu is an Associate Professor of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, China. He obtained his PhD degree from the Waseda University, Japan. He was a Postdoctoral Researcher for the Research Institute for Secure Systems (RISEC), National Institute of Advanced Industrial Science and Technology (AIST), Japan. He is the Chair of IEEE P21451-1-5 Standard Working Group. He is the Associate Editor of *IEEE Access*, Guest Editor of *IEEE Sensors Journal* and TPC member of more than ten international conferences including *ICC*, *GLOBECOM*, etc. His research interests include the fog computing, software-defined networks (SDN), social networks, etc.

Jianhua Li is a Professor/PhD Supervisor and the Dean of the School of Cyber Security, Shanghai Jiao Tong University, Shanghai, China. He is also the Director of National Engineering Laboratory for Information Content Analysis Technology and the Director of Shanghai Key Laboratory of Integrated Administration Technologies for Information Security, China. He was the Chief Expert in the Information Security Committee Experts of National High Technology Research and Development Program of China (863 Program) of China. He has published more than 300 papers. His research interests include network security, cloud computing, big data, etc.

Gaolei Li received his BS degree in Electronic Information Engineering from the Sichuan University, Chengdu, China in 2015 and he is pursuing his PhD degree in the School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai, China. His research interests are focusing on services optimization for internet of energy, fog computing, software-defined networks, etc. He is a TPC member of *International Conference on Internet of Things (iThings 2017)*. He is a student member of IEEE.

This paper is a revised and expanded version of a paper entitled 'An instant message scheme for cross-domain SDN controllers in cloud data center' presented at The International Conference on Smart X, Dalian, China, 30 July 2016.

1 Introduction

Based on a hierarchical approach, software defined networks (SDN) separate the control plane from the data plane. In control plane, there is a programmable controller which includes all of control modules, such as link discovery and topology management. In data plane, plenty of SDN switches, which only take charge of data forwarding, connect together by physical links. OpenFlow (McKeown et al., 2008; Lee et al., 2016; Li et al., 2015, 2016; Wang et al., 2017; Benamrane et al., 2016), as a communication protocol between the SDN controller and switch, plays a critical important role in SDN networks. Through OpenFlow, a SDN controller (Doriguzzi-Corin et al., 2016; Luo et al., 2015; Izard et al., 2016; Yang et al., 2017; Caba and Soler, 2016; Alsmadi, 2016; Gude et al., 2008; Koponen et al., 2010; Yeganeh and Ganjali, 2012; Tootoonchian and Ganjali, 2010) can send forwarding policy to a SDN switch and a SDN switch can report its requests to the related SDN controller too. Due to the programmable features, a SDN controller also calls network operate system (NOS). A SDN controller is extensible that we can flexibly add some function modules into it.

With the increase of network size, many companies build data centres in different regions. There is a tendency for companies to update their network architecture from traditional TCP/IP to SDN. A data centre is always deployed in a place far away from the others, so that it is quite necessary for different data centres to own their personal SDN controllers to ensure the quality of service (QoS).

In this paper, we treat a data centre which takes advantage of SDN technology as a SDN domain. Therefore, different data centres of a company will form a distributed SDN control plane architecture. In this paper, a cross-domain load balancing mechanism based on Extensive Messaging and Presence Protocol (XMPP) (Farkas et al., 2015; Guo et al., 2016; Guo et al., 2015a, 2015b) with push model for flat distributed SDN control plane of enterprise network is proposed. The proposed scheme relies on XMPP using push functionality for achieving better scalability and shorter delay. As the instant message character of XMPP, our proposed approach makes it possible for a controller to send its updated state information to other controllers in real-time. The updated state information in the proposed mechanism is mainly associated with the load of SDN controllers (Lange et al., 2015; Aslan and Matrawy, 2016; Frosterus et al., 2015; Tuncer et al., 2015; Gutierrez-Garcia and

Ramirez-Nafarrate, 2015; Sanjeevi and Viswanathan, 2017; Kessl, 2016), Therefore, the instant message mechanism can solve a single point of failure to some extent by providing the real-time updated state information to load balancer.

The rest of paper is organised as follows: we present the related work in Section 2. Then we describe the basic architecture in Section 3 and analyse the communication mechanism in Section 4. We do some analysis in Section 5. Then we conclude this paper in Section 6.

2 Related work

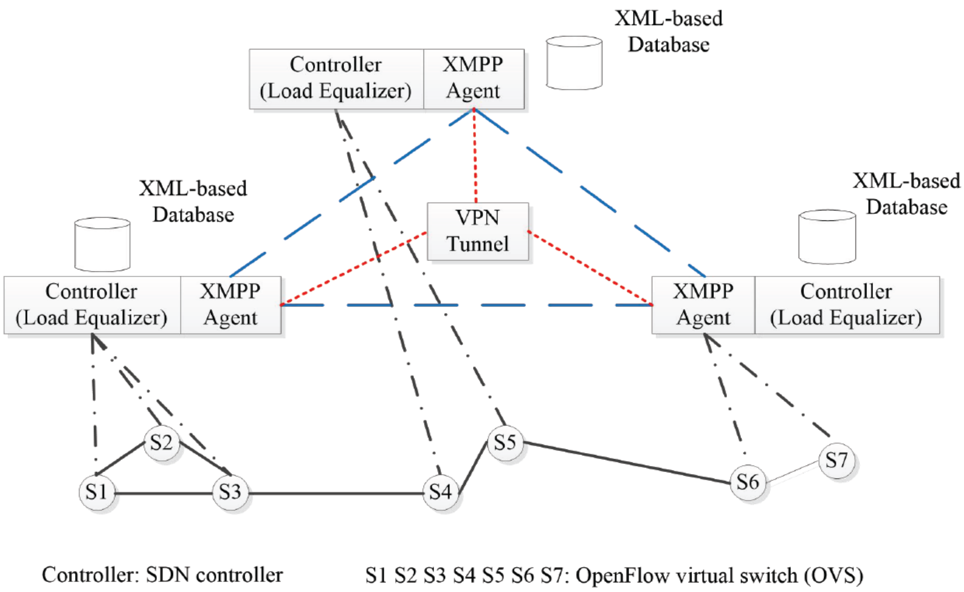
Several attempts have been done to distribute SDN controllers. ElastiCon (Dixit et al., 2014) presents an elastic distributed controller architecture which balances the load across controllers thus ensuring good performance at all times while a WE-Bridge mechanism to enable different SDN administrative domains to peer and cooperate is proposed in Lin et al. (2015). Samadi et al. (2015) presents a novel converged data centre network architecture based on SDN technology to enable on-demand rack-to-rack connectivity across data centres and Phemius et al. (2014) analyses an extensible distributed SDN control plane which can manage its own network domain and communicates with other controllers to provide end-to-end network services thus coping with the distributed and heterogeneous nature of modern overlay networks. A novel converged inter/intra data centre network architecture was presented in Samadi et al. (2015). The architecture enables physical distributed data centres connect rack-to-rack on demand while Casellas et al. (2015) shows a distributed SDN control plane which makes use of multi-protocol label switching protocols (GMPLS) as their east/west interfaces. In addition to the related technology of XMPP, ISMF (Si et al., 2013) shows an example of how to simplify cloud server management and enhance scalability and flexibility in large-scale data centre through the push mechanism (Liaskos and Tsioliaridou, 2016) of XMPP. The active XML rules enable servers to send relevant information to interested remote users in response to new events occurring in the XML-based repositories.

In this paper, we propose an instant messaging scheme for load balancing among different cloud data centres based on XMPP with push model. We combine the extensible features of XMPP with the active push services provided by XMPP extensions to enable one SDN controller to send its state information, like cpu, memory, network bandwidth, to other interested controllers in real time. Compared to other articles, our scheme takes advantage of the instant message and active push character of XMPP to solve the problem of non-real time information synchronisation among multiple controllers in the flat distributed SDN control plane. The load module in SDN controllers can achieve load balancing based on the real-time consistent state information. Thus, our proposed mechanism can solve the problem of a single point of failure in cloud data centre to some extent.

3 Basic architecture

We propose an instant messaging scheme for flat SDN control plane based on XMPP. The proposed scheme enables a SDN controller to push its updated information to other domains in real time. The updated information in this paper is mainly the state information of SDN controllers, such as the usage of cpu, bandwidth utilisation, network overhead, etc. Compared to other distributed systems based on pull technology, it overcomes the shortcomings of non-real time information synchronisation. The realisation of much open source software about XMPP makes our instant messaging scheme very easy to implement in the future.

Figure 1 A cross-domain load balancing scheme for distributed controller plane (see online version for colours)



As described in Figure 1, the control plane includes five main parts: SDN controller, XMPP agent, XML-based database, OpenFlow virtual switch (OVS), load equaliser and virtual private network (VPN) tunnel. In this section we mainly analysis SDN controller and VPN tunnel. Other modules will be described in the following sections.

3.1 SDN controller

The SDN controller plays a big role in the SDN networks which contains a collection of pluggable modules, like, link discovery, switch manager and so on. These modules provide different network functions. A SDN controller consists of two parts: an intra-domain part and an inter-domain part. An intra-domain part takes charge of managing topology, switches, hosts, flow table of the local domain. The central component in an intra-domain part is network information-base (NIB) which holds a collection of network entities, each of which comprises of a set of key-value pairs. An inter-domain part is used for exchanging global-wide network information with other controllers. In our instant messaging system, we integrate a XMPP agent module into every SDN controller. As NIB is mainly used for storing the information of local domain, we add a XML-based database into every SDN controller for storing the global-view information. The XML-based database supports certain operations, like add, update, delete and query based on Extensible Markup Language (XML) related technology. The XMPP agent plus with XML-based database enables a controller to communicate with other controllers in real time.

3.2 VPN tunnel

For the sake of security, almost all large enterprises utilise VPN tunnel technology to achieve the communication among different cloud data centres located in different locations. VPN tunnel is constructed by establishing a logic point-to-point network connection through the use of traffic encryption and virtual tunnel protocol. Therefore, in this paper, we adopt VPN tunnel technology for secure transmission of XML data packets in the backbone network. We just need to encapsulate the XML data packets at the edge of one certain cloud-based data centre and prepare for decapsulating the related XML data packets at the edge of the other data centre. As XML data packets are mainly the state information of SDN controllers, such as the usage of cpu, memory, network, there is no need to consider the effects of encapsulating or decapsulating XML data packets.

4 Proposed instant message scheme

To understand the instant message mechanism of flat distributed SDN control plane, we not only need to discuss the functionality of every component but also have the obligation to analyse the communication process among different components.

4.1 Component design

The instant messaging system consists of the following components. For the decentralised character of XMPP, we only discuss the situation that a controller pushes the changed information to other subscribed controllers. The relationship of the components is described in Figure 2.

1 XML-based database is used for storing global-view load information of SDN controllers. The global-view load information is a collection of network entities, each of which contains a set of key-value pairs.

2 XMPP agent consists of XMPP servers and XMPP client. XMPP servers are responsible for pushing real-time updated load information to other interested controllers while XMPP client supports subscribing load information from others. It is integrated into all the controllers.

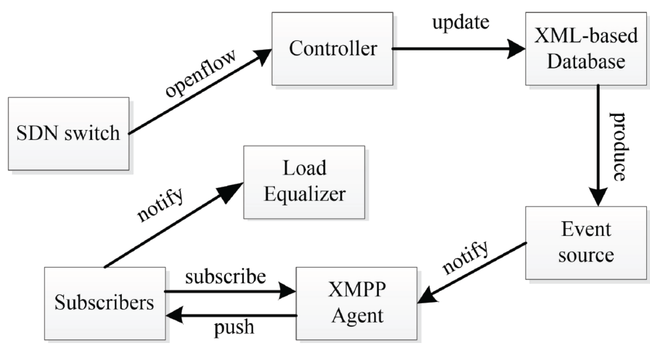
3 Controller module is a SDN controller in the local domain which integrates with XMPP agent. All SDN controllers in our scheme are connected through vpn tunnel.

4 SDN switch module is a plenty of switches which support OpenFlow protocol. The switch can be physical switch orvirtual switch, like OpenFlow virtual switch (OVS). In this paper, the SDN switch mainly refers to OVS.

5 Event source is triggered by XML-based database module. In this paper, it can generate a variety of events, each corresponding to certain load updated information of controllers. Thereafter, it will notify XMPP agent about the changing information.

6 Load equaliser is an important module in SDN controllers. Based on the real-time updated state information by XMPP server, it can achieve better load balancing among SDN controllers.

Figure 2 The components of instant message system



4.2 Communication mechanism

The instant message system is based on push mechanism with a rule-based model. Other controllers subscribe to the rules that they care about from XMPP agent. When the datasets of database in the local controller change, they can fetch these updated data matching the design rule repository in real time and synchronise these data into their own database.

The system uses XMPP and OpenFlow protocol together. As we all know, the OpenFlow protocol is used for the communication between SDN controller and SDN switch while XMPP is responsible for coordinating the communication of all other components in our system. We take advantage of these two protocols together to enable our system to become more reliability and stability.

The communication process of the system is described in Figure 3.

1 Subscriber module firstly subscribes to certain rules from the XMPP agent module. Then the module will send a message representing the successful subscription to subscriber module.

2 OpenFlow switch sends a request to the related controller. Then the controller gives it a response through OpenFlow protocol. This process is likely to update the NIB of the controller. If the dataset of NIB is updated, the controller needs to store the changed information to the XML-based replicated database.

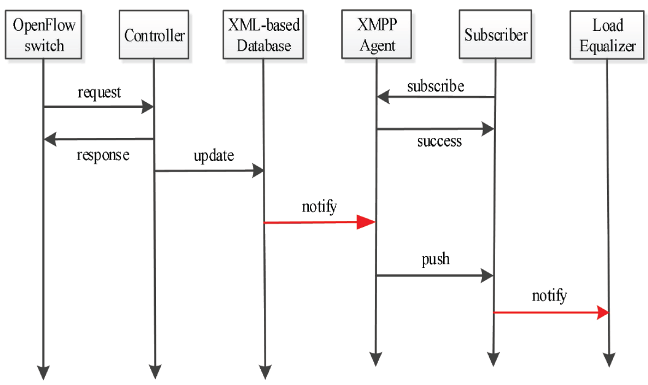
3 A trigger module in the XML-based database will notify the event source when the dataset of the XML-based database is updated.

4 The event source creates an event and publishes it on the specific topic which stores in the repository of XMPP server.

5 The XMPP server makes use of push technology to send the updated event to specific controllers that subscribe the related topic.

6 Load equaliser takes some measures to achieve better load balancing among controllers in cloud data centre based on the real-time state information.

Figure 3 The communication process of the system (see online version for colours)



4.3 Rule repository

The rule repository consists of a set of entries and each of them is similar to a key-value pair. There are three types of elements in an entry: identification, event and action. The identification element is used to show the uniqueness of an entry so that different entries must have different identification. The event element comprises a set of key-value pairs. A key represents an attribute while a value is corresponding to the value of related attribute. The action element consists of a series of action sets like ADD, DELETE and UPDATE. Among these three elements, the identification and event element is required while the action element is optional.

Figure 4 The rule for switch

C	S	action

C	Origine	Update	UPDATE

C1	C2	S	MIGRATE

C,C1,C2: controller ID
S,Origine,Update: switch ID
action: ADD DELETE

As described in Figure 4, the rule for switches is used for checking the state of SDN switch which consists of three elements except for the UPDATE, MIGRATE action. The three elements are a controller id, a switch id and a corresponding action. A controller id is a 48 bit MAC address while a switch id is a 32 bit IP address. A corresponding action is a mapping between SDN controller and SDN switch. The ADD action indicates that the SDN switch is added to a special domain managed by the SDN controller while the DELETE action suggests that the SDN switch is removed from the domain. On the contrary, there are four elements in the UPDATE action: a controller id, a updated switch id, an original switch id, a UPDATE action. The meaning of these elements is as same as the discussed before so that The UPDATE action represents that a new SDN switch (usually a virtual switch in the cloud-based data centre) takes the place of the original switch. The MIGRATE action also consists of four elements: C1, C2, a switch id and the MIGRATE action. C1 is the id of original controller before migration while C2 is the id of new controller after migration. With this MIGRATE action entry, it enables us to know the migration of switch in real time.

The other kind of entry is used to calculate the load of a local SDN controller. In our instant message system, we use four parameters to measure the load of a server. These four parameters are cpu, memory, disk and network bandwidth. As described in Figure 5, this type entry comprises of five elements. There are a controller id, the usage of cpu, the usage of memory, the state of disk and the bandwidth of network. The usage of cpu, memory, disk and network can be acquired by certain tools or a simple python script.

Figure 5 The rule for controller

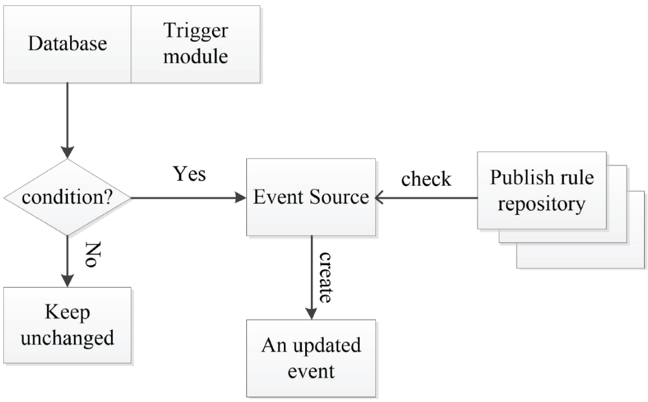
C	CPU	Memory	Disk	Network	Count

4.4 XML-based database and event source

As a complement of NIB of SDN, the XML-based database plays a big role in our instant message system. First of all, the XML-based database needs to be initialised by the content of NIB. And if the request of SDN switch changes the datasets of NIB, the XML-based database should immediately update its content according to these changes. There is a trigger module in the XML-based database. The trigger module is used for notifying the event source when the dataset in the XML-based database changes. To find out what datasets have been changed, the trigger module is responsible for comparing the content before and after modification. As the XML-based database communicates with other modules through XMPP protocol and the XMPP protocol is based on XML, it is quite necessary for the database to convert the query results to XML format. Then the other modules can parse the results by the use of XML-related technology, like XQuery.

The communication between XML-based database and event source is described in Figure 6. Firstly, the XML-based database makes use of the trigger module to query whether its data changed. If the database is different after modification, a notification will be sent to event source. Finally, the event source is responsible for creating an updated event according to the publish rule repository.

Figure 6 The communication between replicated database and event

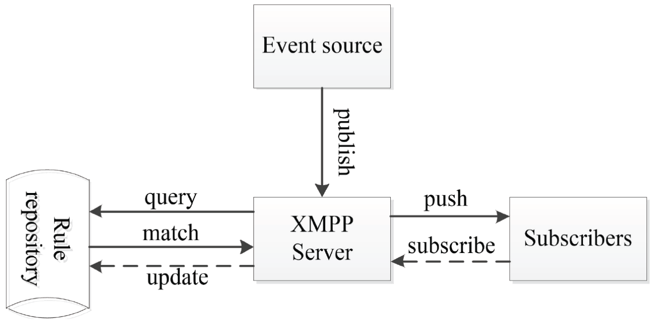


4.5 XMPP server and subscriber

As discussed above, XMPP server is a core part of XMPP agent module. We can see from Figure 7 that the XMPP server has two things to do. One is receiving the request of subscribers, updating the rule repository according to the subscription. The other is querying the rule repository when the event source produces an event notification on certain rules. The rule repository is a XML-based database which supports operation by using certain XML technology, like XQuery for query. If results of this query match with subscribed rules, the XMPP server makes use of push functionality to send the event notification to the corresponding subscribers.

In a cloud-based data centre, the topology and network traffic change constantly which is very likely to cause load unbalancing among different domains. We can take advantage of the pushing character of XMPP server to enable a controller to learn the load information of other controllers in real time. Therefore, load module of each controller can take some measures to achieve load balance in data centres according to the global-view load information of SDN controllers.

Figure 7 The communication between XMPP server and subscriber

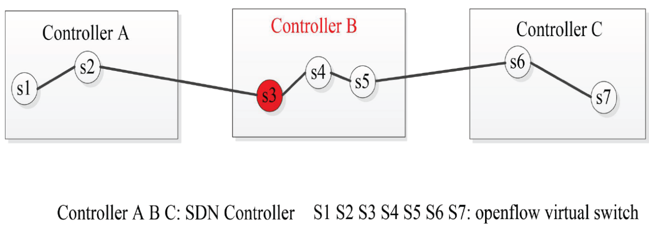


5 Analysis and evaluation

In an enterprise data centre, the network traffic change dynamically. It is very likely that the load of a domain is heavy while another domain takes charge of small network traffic. Finally, it may cause that the load among controllers is not balanced which can lead to some serious problems, like, a single point of failure, server failure. However, our instant message scheme has the ability to solve these problems to a certain extent because it can push updated state information of one controller to other interested controllers in real time. In our proposed mechanism, the pushed state information is mainly the load of controllers, such as the usage of cpu, memory or network overhead. These updated state information is vital to load equaliser in the SDN controller as load equaliser can make use of these information to achieve load balancing among SDN controllers. Thus, our proposed mechanism makes a great contribution to optimise the allocation of resources among SDN controllers.

As discussed above, we define two kinds of rules for event notification. The rule for switch is used for monitoring the relation between switch and controller while the rule for controller enables network administrator to know the load information of all controller nodes. We can make use of the rule for controllers to monitor the load state of any controller. The load state comprises the usage of cpu, memory, disk, network bandwidth as well as the number of managed switches. When the load gap between two controller nodes exceeds a certain value, it reminds the administrator to take appropriate measures to dispatch some tasks from a heavy controller to a light controller.

Figure 8 Load monitor and schedule of the proposed scheme (see online version for colours)

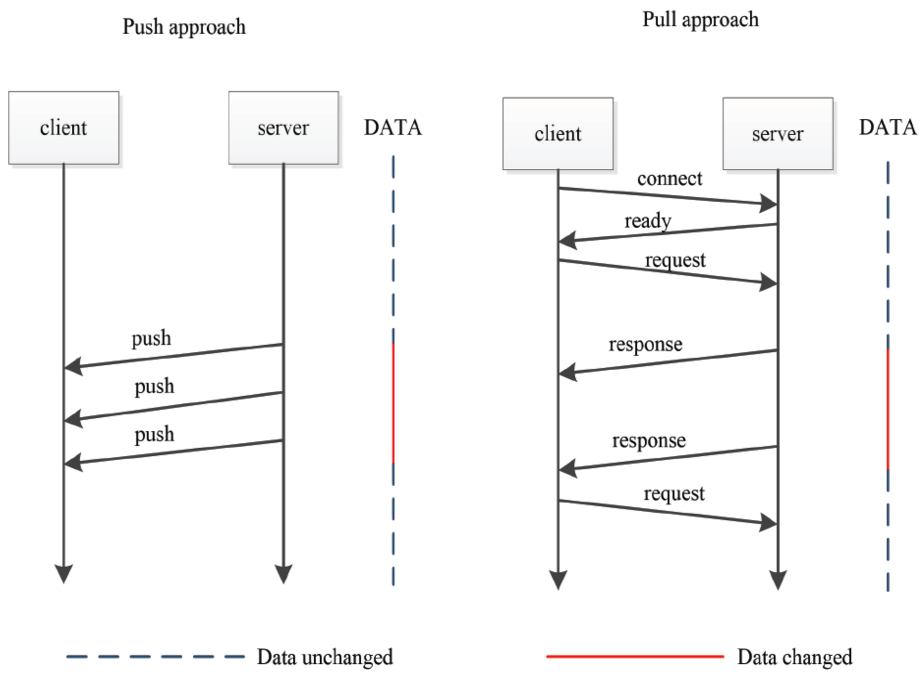


The most common way to lighten the load is switch migration. In a cloud-based data centre, there are two kinds of switch: physical switch and virtual switch. The number of virtual switches is much more than the physical switches and virtual switch supports OpenFlow protocol better than physical switch. In this paper, we only consider the migration of virtual switch. As described in Figure 8, we suppose that an enterprise consists of three data centres which located in different areas. Each data centre is managed by a SDN controller cluster and all of the switches are OVS. We assume that the domain managed by controller cluster B is under the heavy load while controller A remains large amounts of cpu resources and controller C has adequate network bandwidth resources according to the result of load monitoring. Through network analysis tool, we find that S3 takes up a lot of network bandwidth and cpu resource. Therefore, we elect S3 as the migration node. We can transfer S3 to controller A or controller C according to resource requirements and load balancing algorithm. Finally, this change will be pushed to other controllers based on the rule for switch.

There are two ways for exchanging information between two network nodes. A poll-based approach is based on request response mechanism while a push-based approach makes use of publish-subscribe mechanism. We can see from Figure 9, a client needs to connect to a server before it can receive message from the server in a poll-based approach. If the server is ready for this connection, it sends a response to the client. After receiving the ready response from the server, the client can send a request to fetch certain data. The connection between client and server will not stop until the data of server is ready. Different from the pull-based approach, a push-based approach does not need to maintain a long connection between a client and a server. The communication is initiated by the server and the server knows nothing about client nodes. The server just pushes an event notification to client nodes when the subscribed dataset is changed. We can conclude that a poll-based approach consumes more computing and network resources while a push-based approach needs to add an intermediate node to forward an event notification. Therefore, a poll-based approach is better suited for stable environment while a push-based approach is suited for rapidly changing data environment.

In this article, we use push model provided by XMPP publish/subscribe extension to synchronise the global-view information to all the controllers. The push model overcomes the shortcomings of poll model and it has the ability to synchronise the rapidly changing global information to all the controllers in real time. Therefore, it is well suited for the flat SDN control plane of enterprise network which consists of a set of data centres. Next, we use Mininet simulation tools to do some experiments to verify the advantage of the proposed mechanism. The first one is for testing the packet delay of end-to-end communication. The second one is used for testing the packet loss. The last one is used for testing the network throughput. The packet delay is an vital import index measuring the speed of pushing updated information by XMPP server. In this paper, we do not consider tunnelling delay because it is a common and mature technology used in cloud centres. The faster the speed of information transmission, the more likely providing information to the load module timely. Thus, it enables cloud data centres to achieve better load balancing results and avoid a single point of failure. The second important index is packet loss which is a measure of network overhead. The last index is network throughput.

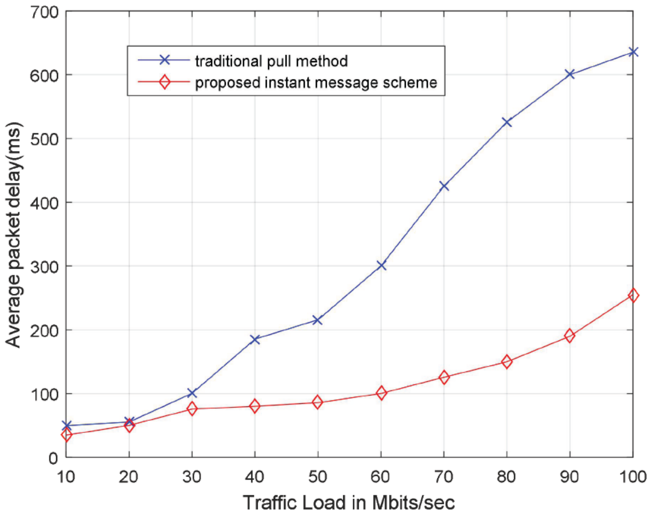
Figure 9 The mechanism of push and pull scheme (see online version for colours)



5.1 Packet delay

Figure 10 depicts the average packet delay for both our proposed instant message scheme and traditional pull approach. We can see that the latency of the two schemes is basically the same when the network traffic is between 10 M and 30 M. With the rise of network traffic, the latency of traditional pull method growth rapidly while the latency of our proposed method increases slowly. A careful observation of Figure 10 shows that the maximum delay of our push approach is about 260 ms while the pull approach is up to 620 ms. Therefore, we can conclude that the average latency of our proposed approach is better than the pull approach according to the result of simulation.

Figure 10 The average packet delay of network (see online version for colours)

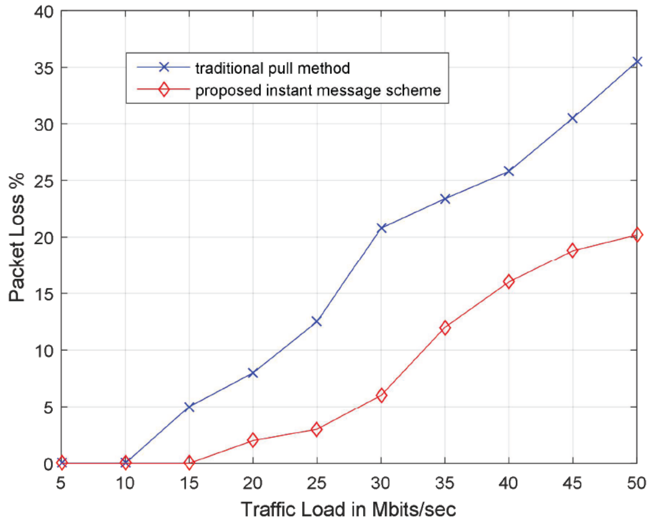


The latency performance of our proposed method is better than pull approach because pull method needs to continually send heartbeat message to monitor the state of the certain controllers. When the state of servers have changed, the servers response to the certain clients(generally other domain controllers) its changed message in order to maintain the consistency of the global network view. Moreover, like e-mail, when one XMPP client sends an XMPP message to one of its contacts at a different domain, the XMPP client firstly establish a connection with its server which then connects directly the contacts server without intermediate hops. At last, our push mechanism not only reduces the communication overhead of the controller cluster, but also shortens the time delay which is quite important to distributed network environment.

5.2 Packet loss

Figure 11 shows the average packet loss rate of the network. In the initial stage, the network traffic is small, and the average packet loss rates of both schemes are around zero. However, the number of packets through the underlying network has increased significantly over time resulting in increased packet loss rate of both schemes. The growth rate of our proposed instant message scheme is less than traditional pull approach. The reason for this phenomenon is that our proposed scheme is based on the push mechanism. Compared to the traditional method, this scheme can reduce network overhead and network congestion under large-scale network traffic.

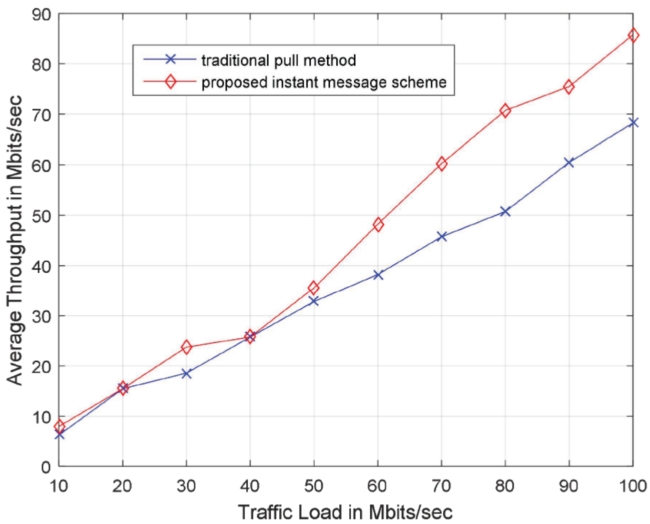
Figure 11 The average packet loss rate of network (see online version for colours)



5.3 Network throughput

Figure 12 depicts the average throughput of our proposed instant message scheme based on XMPP related technology and the traditional pull approach. When the size of network traffic is small, the network throughput of the two schemes is basically the same. With the increase of network traffic, the average throughput of our proposed scheme is great than the traditional method, we can attribute this phenomenon to lower packet loss and lower network bandwidth consumption.

Figure 12 The average throughput of network (see online version for colours)



6 Conclusions

In this paper, we propose a cross-domain load balancing scheme based on XMPP and XMPP push model for flat distributed SDN control plane of enterprise network. The proposed scheme relies on push technology which is supported by XMPP publish/subscribe extension enabling all the controller nodes to share the same global-view load information. We also take advantage of XMPP core functionality to enable a controller to communicate with other controllers by XML stream. Our cross-domain load balancing scheme has the ability to synchronise the updated information of a domain to other controllers in real time. The real-time consistent state information enable load equaliser in SDN controllers to achieve load balancing and avoid a single point of failure.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China under Grant 61431008, 61571300 and partially supported by the JSPS KAKENHI Grant Number JP16K00117, JP15K15976, KDDI Foundation.

References

Alsmadi, I. (2016) 'The integration of access control levels based on SDN', *Int. J. of High Performance Computing and Networking*, Vol. 9, No. 4, pp.281–290.

Aslan, M. and Matrawy, A. (2016) 'On the impact of network state collection on the performance of SDN applications', *IEEE Communications Letters*, Vol. 20, No. 1, pp.5–8.

Benamrane, F., Ros, F. and Mamoun, M.B. (2016) 'Synchronisation cost of multi-controller deployments in software-defined networks', *Int. J. of High Performance Computing and Networking*, Vol. 9, No. 4, pp.291–298.

Caba, C. and Soler, J. (2016) 'Mitigating the controller performance bottlenecks in software defined networks', *Int. J. of Communication Networks and Distributed Systems*, Vol. 17, No. 3, pp.275–295.

Casellas, R., Martínez, R., Muñoz, R., Vilalta, R. and Liu, L. (2015) 'Control and orchestration of multidomain optical networks with GMPLS as inter-SDM controller communication', *Journal of Optical Communications and Networking*, Vol. 7, No. 11, pp.46–54.

Dixit, A., Hao, F., Mukherjee, S., Lakshman, T. and Kompella, R. (2014) 'ElastiCon: an elastic distributed SDN controller', *The Tenth ACM/IEEE Symposium on Architectures for Networking and Communications Systems*, pp.17–28.

Doriguzzi-Corin, R., Salvadori, E. and Gerola, M. (2016) 'Towards an automated framework to instantiate virtual networks in OpenFlow-based infrastructures', *Int. J. of Communication Networks and Distributed Systems*, Vol. 17, No. 3, pp.234–253.

Farkas, K., Feher, G., Benczur, A. and Sidlo, C. (2015) 'Crowdsending based public transport information service in smart cities', *IEEE Communications Magazine*, Vol. 53, No. 8, pp.158–165.

Frosterus, M., Tuominen, J., Pessala, S. and Hyvonen, E. (2015) 'Linked open ontology cloud: managing a system of interlinked cross-domain lightweight ontologies', *Int. J. of Metadata, Semantics and Ontologies*, Vol. 10, No. 3, pp.189–201.

Gude, N., Koponen, T., Pettit, J., Pfaff, B., Casado, M., McKeown, N. and Shenker, S. (2008) 'Nox: towards an operating system for networks', *ACM SIGCOMM Computer Communication Review*, Vol. 38, No. 3, pp.105–110.

Guo, L., Dong, M., Ota, K., Wu, J. and Li, J. (2015a) 'Event-oriented dynamic security service for demand response in smart grid employing mobile networks', *China Communication*, Vol. 12, No. 12, pp.63–75.

Guo, L., Wu, J., Xia, Z. and Li, J. (2015b) 'Proposed security mechanism for XMPP-based communications of ISO/IEC/IEEE 21451 sensor networks', *IEEE Sensors Journal*, May, Vol. 15, No. 5, pp.2577–2586.

Guo, L., Dong, M., Ota, K., Wu, J. and Li, J. (2016) 'A name-based secure communication mechanism for smart grid employing wireless networks', *IEEE GLOBECOM*.

Gutierrez-Garcia, J. and Ramirez-Nafarrate, A. (2015) 'Collaborative agents for distributed load management in cloud data centers using live migration of virtual machines', *IEEE Transactions on Services Computing*, Vol. 8, No. 6, pp.916–929.

Izard, R., Deng, J., Wang, Q., Xu, K. and Wang, K.C. (2016) 'An agent-based framework for production software defined networks', *Int. J. Communication Networks and Distributed Systems*, Vol. 17, No. 3, pp.254–274.

Kessl, R. (2016) 'Probabilistic static load-balancing of parallel mining of frequent sequences', *IEEE Transactions on Knowledge and Data Engineering*', Vol. 28, No. 5, pp.1299–1311.

Koponen, T., Casado, M., Gude, N., Stribling, J., Poutievski, L., Zhu, M., Ramanathan, R., Iwata, Y., Inoue, H., Hama, T. and Shenker, S. (2010) 'Onix: a distributed control platform for large-scale production networks', *Usenix Conference on Operating Systems Design & Implementation*, Savannah, GA, pp.351–364, 2010.

Lange, S., Gebert, S., Zinner, T., Tran-Gia, P., Hock, D., Jarschel, M. and Hoffmann, M. (2015) 'Heuristic approaches to the controller placement problem in large scale SDN networks', *IEEE Transactions on Network and Service Management*, Vol. 12, No. 1, pp.4–17.

Lee, S.S.W., Li, K. and Wu, M. (2016) 'Design and Implementation of a GPON-based virtual OpenFlow-enabled SDN switch', *Journal of Lightwave Technology*, Vol. 34, No. 10, pp.2552–2561.

Li, D., Shang, Y., He, W. and Chen, C, (2015) 'EXR: greening data center network with software defined exclusive routing', *IEEE Transactions on Computers*, Vol. 64, No. 9, pp.2534–2544.

Li, G., Dong, M., Ota, K., Wu, J., Li, J. and Ye, T. (2016) 'Deep packet inspection based application-aware traffic control for software defined networks', *IEEE Globecom*.

Liaskos, C. and Tsioliaridou, A. (2016) 'Service ratio-optimal, content coherence-aware data push systems', *ACM Transactions on Management Information Systems*, Vol. 6, No. 4.

Lin, P., Bi, J., Wolff, S., Wang, Y., Xu, A., Chen, Z., Hu, H. and Lin, Y. (2015) 'A west-east bridge based SDN inter-domain testbed', *IEEE Communications Magazine*, Vol. 53, No. 2, pp.190–197.

Luo, S., Dong, M., Ota, K., Wu, J. and Li, J. (2015) 'A security assessment mechanism for software-defined networking-based mobile networks', *Sensors*, Vol. 15, No. 2, pp.31843–31858.

McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, S. and Turner, J. (2008) 'OpenFlow: enabling innovation in campus networks', *ACMSIG COMM*.

Phemius, K., Bouet, M. and Leguay, J. (2014) 'DISCO: distributed multi-domain SDN controllers', *Network Operations and Management Symposium*, pp.1–4.

Samadi, P., Xu, J., Wen, K., Guan, H., Li, Z. and Bergman, K. (2015) 'Experimental demonstration of converged inter/intra datacenter network architecture', *The 17th International Conference on Transparent Optical Networks*, pp.1–4.

Samadi, P., Xu, J., Wen, K., Guan, H., Li, Z. and Bergman, K. (2015) 'Experimental demonstration of converged inter/intra datacenter network architecture', *The 17th International Conference on Transparent Optical Networks*, pp.1–4.

Sanjeevi, P. and Viswanathan, P. (2017) 'Workload consolidation techniques to optimise energy in cloud: review', *Int. J. of Internet Protocol Technology*, Vol. 10, No. 2, pp.115–125.

Si, P., Song, C. and Zhou, X. (2013) 'Intelligent server management framework over extensible messaging and presence protocol', *China Communications*, Vol. 10, No. 5, pp.128–136.

Tootoonchian, A. and Ganjali, Y. (2010) 'Hyperflow: a distributed control plane for openflow', *INM/WREN*.

Tuncer, D., Charalambides, M., Clayman, S. and Pavlou, G. (2015) 'Adaptive resource management and control in software defined networks', *IEEE Transactions on Network and Service Management*, Vol. 12, No. 1, pp.18–33.

Wang, W., Wang, B., Hu, Y. and Liu, B. (2017) 'ReconsMap: a reliable controller-switch mapping in software defined networks', *Int. J. of High Performance Computing and Networking*, Vol. 10, No. 3, pp.201–206.

Yang, C., Liu, J., Chen, W., Leu, F. and Chu, W. (2017) 'Implementation of a virtual switch monitoring system using OpenFlow on cloud', *Int. J. of Ad Hoc and Ubiquitous Computing*, Vol. 24, No. 3, pp.162–172.

Yeganeh, S.H. and Ganjali, Y. (2012) 'Kandoo: a framework for efficient and scalableoff loading of control applications', *HotSDN*.