Software-based Fast Failure Recovery in Load Balanced SDN-based Datacenter Networks

Behrouz Raeisi
Department of Computer Science
University of Pisa / Scuola Superiore Sant'Anna
Pisa, Italy
e-mail: b.raeisigahrouei@unipi.it

Alessio Giorgetti
Institute of Communication, Information and Perception Technologies
Scuola Superiore Sant'Anna
Pisa, Italy
e-mail: alessio.giorgetti@sssup.it

***Abstract*—Load balancing is currently considered as a candidate solution to tackle the emerging problem of increasing bandwidth demand in intra-datacenter networks. Furthermore, because a short disruption of data transfer would corrupt the result of a long procedure of computation, fast failure management mechanisms are considered as integral part of current datacenters. In this paper, we propose a method which uses active probing to detect and manage failures in an OpenFlow based datacenter network exploiting load balancing among equal cost multiple paths. The proposed method is scalable and effective based on the actions it takes without involving the controller in the fast failure recovery procedure.**

Keywords-SDN; fast failure recovery; load balancing; probe; openflow

I. INTRODUCTION

In an attempt to increase the network flexibility and tractability [1], SDN (Software Defined Networking) was born through academic circles. Thereinafter, due to lower CAPEX and OPEX [2], it was noticed by large providers as a powerful and efficient architecture for carrier grade networks. In fact, the main idea is to decouple control plane from forwarding plane and transfer it to a central controller. By doing so, the controller will have a comprehensive view of network and can implement and deploy complex network management scenarios, such as failure recovery, and load balancing.

However, similar to other emerging technologies, SDN has several major drawbacks; one of which is possibility of controller saturation [3] because of numerous requests and statistical data it receives in large networks. Accordingly, any effort for implementing the aforementioned scenarios must be performed by considering this constrain.

Nowadays, various forms of SDN are emerging for carrier grade networks including SR (Segment Routing) [4], while the most traditional SDN using OpenFlow [5] protocol is suitable for intra-datacenter networks. Moreover, the drift of providers to preserve and deploy the traditional network architecture based on independent network devices consist of data and forwarding planes is another obstacle toward expansion of SDN in carrier based networks. Accordingly, the focus of this research is more shifted toward SDN implementations in datacenters in comparison to the carrier networks. Nevertheless, ample cases of SDN based networks have been deployed [6], [7], [8], [9] in large datacenters and carrier networks.

The switch communication with the controller in the SDN network [10] could be done through in-band or out-of-band connections (Fig.1). The benefit of in-band communication is reduction of ports required to communicate with the controller, which is preferred by some providers, while any failure, particularly in the controller edge switch, would disrupt the connection with the controller and as the result execution of any failure recovery mechanism. Nevertheless, because plenty of high speed ports are available in modern datacenters, out-of-band communication with the controller is the dominant method in datacenters.

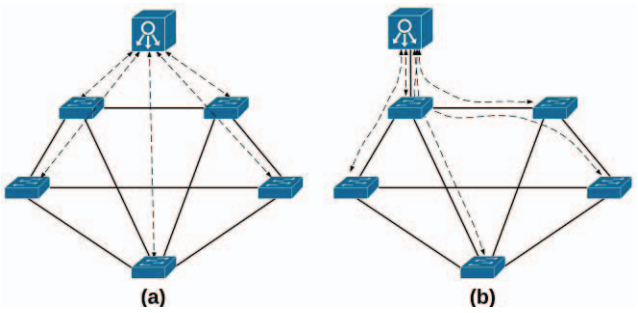

Figure 1. out-of-band (a) and in-band (b) controller communication

There are various studies using probes, both active and passive, for network monitoring purpose [10], [11], [12], [13], [14], [15]. Nevertheless, active probes have several advantages over passive monitoring techniques [16], such as (i) less instrumentation (ii) capability to compute end-to-end performance (iii) quicker localization, etc. Furthermore, two major problems in probe based monitoring are probe station selection and probe set selection [16], [17]; the former addresses the problem of selecting minimum subset of nodes in the managed network where probe station must be placed such a way that best monitoring can be achieved. The latter defines probe set in such a way that all of elements are probed, minimum probe set is required for network

978-1-5090-3495-6/16/$31.00 ©2016 IEEE

monitoring, and has better performance in bandwidth usage and delay.

In respond to doubling bandwidth demand every 12-15 months in datacenters [8], the intra-datacenter connections speed are started to increase from 10G to 40G recently and to 100G in the immediate future. However, without load balancing to exploit efficiently all of redundant connections in datacenters, few links experience congestions while other links are being underutilized [18], [19]. Consequently, such bandwidth demand cannot be fulfilled even with link speed growth in intra-datacenter networks. Furthermore, load balancing typically enables latency reduction in intra-data center communications.

Having much more data at stake in the modern data centers, also fast failure recovery mechanisms have received more attention recently, avoiding result of sensitive and critical computations being corrupted. Generally, failure management mechanisms can be divided into two different phases, detection and restoration, each of which with its own implementation mechanism and execution time.

This paper tries to investigate and provide an insight into deployment of an effective load balancing mechanism in intra-datacenter networks based on OpenFlow and the impacts of failures on such load balancers, although the results can be expanded for other types of networks. Regarding of failure detection, we exploit active probes send from each Top-of-Rack (ToR) switches and flooded by the aggregate switches (cluster switches) to all other ToR switches of same cluster (Pod). By exploiting this technique, all of ToR switches can perform local configuration modifications and act independently of central controller, avoiding controller saturation and scalability issue.

The reminder of this paper is organized as below. In section II, we discuss related works while in section III, we introduce a load balancing and failure recovery scheme based on select groups in OpenFlow and investigate network behavior in the case of a failure incident. In section IV, we demonstrate our failure scheme exploiting active probes and compare it to the traditional controller based failure management mechanisms. Finally, we finish this paper with conclusion and remarks in section V.

II. Related Works

Lack of academic attention to failure management mechanisms, made the current SDN-based networks to rely on traditional and non-optimal algorithm and technologies for failure management [12]. The following is few available studies, worth of mentioning, about failure management in SDN-based networks.

In some early works, LLDP (Link Layer Discovery Protocol) [20], OpenFlow events [21], and BFD (Bidirectional Forwarding Detection) [22] are used to check the links status and inform the controller upon a failure occurrence in order to manage it. The results depict few hundreds of milliseconds are required for failure detection and restoration. In [23], predefined back-up paths for fast failure recovery are used. Without involving controller, after detecting failure by a switch, an automatic failure recovery is performed, but scalability problem due to large number of installed flows is observed.

Some other works used Netography [24] to investigate packet behavior and pinpoint failure in the network. For example, in [10], [11], a two round procedure is proposed for in-band and out-of-band controller communications. First, in the Path Alive Monitoring (PAM) round, controller sends a monitoring packet through all links in the network. If the whole links are alive, the monitoring packet will eventually come back to the controller. If controller does not receive monitoring packet, a link failure is detected and the detection process goes to Failure Location Identification (FLI) round. In FLI round, controller sends an FLI packet through the links in the network and expects to receive a LFI copy forwarded by the switches in the monitoring path. The node which does not send a LFI copy to the controller is considered adjacent to the failed link. The result by 50 msec monitoring period showed 100 msec for recovery in the worst case.

In [12], a tool named SPIDER is introduced which consists of a stateful data plane based on OpenState [25]. In fact, OpenState pipeline is a legacy OpenFlow flow table preceded by a state table to store “flow states”. Initially, the controller must be provided by the backup paths and deploy detour paths on the switches in boot time. Furthermore, SPIDER uses MPLS tags to distinguish different forwarding behaviors to react upon failure without controller involvement, and bidirectional heartbeat packets for failure detection. Therefore, based on the various tags, packets can be forwarded from detours or the default paths. However, incapability to differentiate between link or switch failure and ample flow entries per nodes, which makes the tool unsuitable for data center switches, are the main drawbacks of SPIDER.

All of aforementioned studies do not focus on a comprehensive solution including both probe-based fast failure recovery schemes and load-balancing among equal cost multiple paths within data center networks.

III. Failure Recovery in a Load Balanced OpenFlow Network

Groups were introduced for the first time in OpenFlow 1.1 [5]. The ability for a flow entry to point to a group enables OpenFlow to represent additional methods of forwarding. Each group is consisted of an ordered list of action buckets, where each action bucket contains a set of actions to execute and associated parameters. Furthermore, the notion of *Liveness* is introduced in OpenFlow specification; providing the OpenFlow switches with the capability to choose group buckets based on availability of other groups or ports. Nevertheless, between different group types, select group can be exploited for load balancing

purpose. According to specification of select group, packets are processed by a single bucket, based on a switch-computed selection algorithm.

By exploiting Liveness feature incorporated in Open-Flow groups, also known as watch group and watch port, OpenFlow switches are able to perform automatic failure recovery. As a matter of fact, if a link becomes unavailable, the adjacent switch automatically reroute the traffic by choosing another output port defined in the group. However, the link failure detection time is different for interfaces with different hardware [10], [11], and dramatically can impact the overall failure recovery time.

To investigate the failure detection time of our available hardware and have some results to compare to outcomes of our final scheme, portion of a datacenter like network consist of equal cost multiple paths by five Open VSwitches [26], acting as ToR switches, and an HP 3800 switch with two OpenFlow instances, acting as aggregate switches, was deployed (Fig.2). Accordingly, several select groups, one for communication with each ToR switch, were installed through Ryu controller [27] and watch port for each bucket was configured.

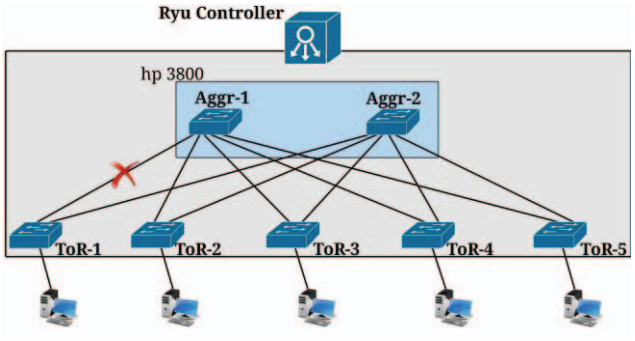

Figure 2. Example of datacenter like network

Furthermore, upon a link failure, all of remote ToR switches must be informed to avoid including the failed link in their communication paths. For example, in Fig.2, if the link connecting ToR-1 to Aggr-1 fails, other ToR switches (ToR-2, 3, 4, 5) must be informed to modify the group using to communicate with ToR-1. Initially, to inform these switches we involved the central controller by exploiting OpenFlow events which in fact are passive probes.

In Table I, the initial results are shown. By comparing the results of physical failure simulation in 100M (Fast Ethernet) and 1G interfaces, a fast failure detection time of around 10ms is typically achieved for 100M interfaces. Instead, a longer detection time of around 350ms is measured for the 1G interface, which is unacceptable for modern datacenter deployments. Moreover, scalability issues may arise given the involvement of the central controller.

TABLE I. INITIAL FAILURE RECOVERY SCHEME RESULTS

NIC(Mbps)	Fail Simulation	Min(ms)	Max(ms)	Avg(ms)
100	Administratively	172	177	175
	Physically	4	11	7
1000	Administratively	170	181	176
	Physically	353	361	355

IV. SOFTWARE BASED FAILURE RECOVERY SCHEME

In order to shorten the failure detection phase, the most time consuming phase of the initial recovery scheme, we decided to use active probes, due to their advantages in comparison with passive probes.

Furthermore, to avoid scalability issue and achieve faster failure recovery, the central controller is not involved in the proposed failure recovery scheme and group modifications are only performed locally by probe stations located on ToR switches. Generally, the goal is to diminish failure recovery time for 1G interfaces to less than 50ms and possibly around 10ms.

Before detailing the scheme, first we have to address two important issues of probe station selection and probe set selection.

A. Probe Station Selection

Because group modifications must be performed locally, each ToR switch must be contained a probe station. Nevertheless, locating probe stations on OpenFlow switches has no contradiction with the idea of SDN to use switches with pure forwarding plane because OpenFlow switches already supports probing, according to specification, and local group modifications can be easily performed by pre-configured instructions set by controller on the switches.

B. Probe Set Selection

To have an overview of links in a cluster, all of ToR switches must receive probes sent by each ToR switch in the same cluster. For example, as shown in Fig. 3, for probe associated to link between ToR-1 and Aggr-1, after ToR-1 forwards the probe packet through its associated link, Aggr-1 floods the probe packet to ToR switches inside the cluster. Furthermore, to reduce the bandwidth exploited for probing, the probe only consist of UDP packet, and each link is distinguished by different UDP port number carried in the header. Furthermore, the overall bandwidth used by probes for each link must be taken into account, which varies based on required precision and speed of the failure detection phase and the number of switches inside the cluster. For example, if probe packets are produced every millisecond with 32 to 48 ToR switch in a cluster [8] and UDP packet size of 54 bytes, 13 to 20 Mbps of each link is occupied, which is negligible with respect to 10G to 40G links of modern intra-datacenter networks.

Finally, the monitoring component of probe stations must be designed meticulously because any flaw in this component dramatically increases false positive or false negative in failure detection phase. The monitoring

component expects to receive at least one probe packet in a predefined interval; if not, it assumes the link is failed and the restoration phase begins consist of sending OpenFlow group modification messages. To have a better understanding, consider a failure in the link connecting ToR-1 to Aggr-1 in Fig.3. Upon the failure, all remote ToR switches (ToR-2, 3, 4, 5) will not receive the probe packets related to the failed link and modify the group using to communicate with ToR-1. On the other hand, ToR-1 will not receive the probe packets, flooded by the Aggr-1 and modifies all of the groups using to communicate with other ToR switches in the cluster.

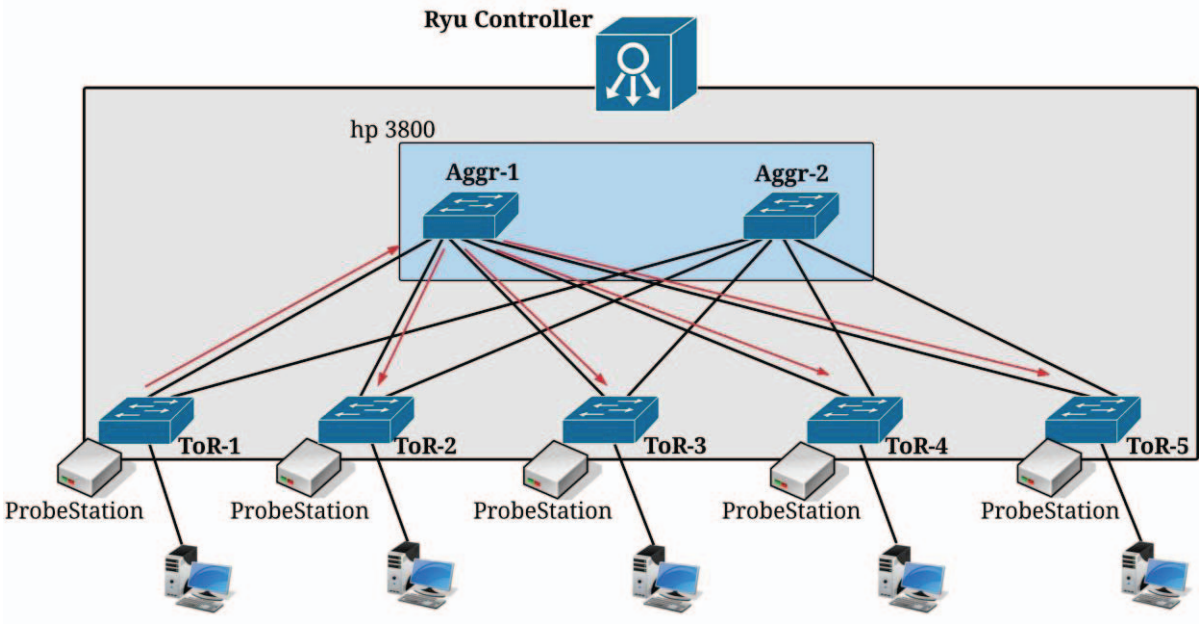

Figure 3. Datacenter like network consist of probe stations located on ToR switches

However, based on our various trials and previous studies [10], [11], the most critical part of monitor component is monitoring interval time. In our trials, we defined two different intervals, one equal to the time required to send two consecutive probe packets and another equal to three consecutive packets. As an illustration, if the rate of packet production is 1000pkt/sec, i.e. one packet every millisecond, the interval is considered 2ms in the first case and 3ms for the second. Nevertheless, during the failure simulation a high level of false positives in the failure detection phase was noticed in the first case, particularly in high link utilization, because of higher packet latency. In addition, even with lower probing rates but the intervals of equal to two consecutive packets transfer, e.g. packet rate of 400pkt/sec and interval of 5ms, same false positives was spotted, although with lower rate.

Furthermore, depending on when in monitoring interval link failure occurs, we expect to spend at least one and at most two monitoring interval for failure detection. Likewise, because probe stations and Open VSwitches are two separate modules, although both are located in the same machine, a short period of time is spent for exchanging OpenFlow messages between two modules. Because of exchanging few group modifications, this time for our test bed is less than 1ms even for the ToR switch connected to the failed link which must perform the largest number of group modifications. However, for larger clusters with 32 to 48 ToR switches, this time can be significant with respect to the short failure detection time. Nevertheless, this message exchange can be eliminated by integrating probe stations to Open VSwitches.

In Table II, the outcomes of our final scheme are shown. Note that trials including false positive are not considered in calculation of the final results. As results depict, fast failure recovery time of less than 10ms is always experienced with probe rate of 1000pkt/sec.

TABLE II. FAST FAILURE RECOVERY SCHEME FINAL RESULTS

Probe Rate(pkt/sec)	Monitoring interval(ms)	Min(ms)	Max(ms)	Avg(ms)
400	5	5	11	7
	7	7.2	13.4	10.5
1000	2	2.2	5.5	3.7
	3	3	7	4.9

V. CONCLUSION AND REMARKS

In this paper, we delved into two indispensible mechanisms of any modern datacenter; first, the load balancing which without it datacenters cannot tackle the problem of increasing bandwidth demands in intra-datacenter network, and second, the fast failure recovery which is vital for datacenters in order to produce consistent data especially in sensitive computations. Furthermore, by finding the failure detection phase as the most time

consuming phase of the initial failure recovery scheme, we improve our recovery mechanism with exploiting active probes. As final results indicate, much faster failure recovery time, several hundreds of milliseconds faster, is achieved by using the proposed scheme. Beside, because the central controller is not involved in the failure recovery procedure, no scalability issue is imposed to the controller.

ACKNOWLEDGMENT

The authors would like to thank Prof. Filippo Cugini of National Interuniversity Consortium for Telecommunications (CNIT) for his valuable suggestions and fruitful discussions.

REFERENCES

[1] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: enabling innovation in campus networks," SIGCOMM, Aug. 2008.

[2] S. Das, G. Parulkar, N. McKeown, P. Singh, D. Getachew, and L. Ong, "Packet and circuit network convergence with OpenFlow," in Optical Fiber Communication Conf. and the Nat. Fiber Optic Engineers Conf. 2010, Mar. 2010, paper OTuG1.

[3] A. Giorgetti, F. Paolucci, F. Cugini, and P. Castoldi, "Dynamic restoration with GMPLS and SDN control plane in elastic optical networks [invited]," J. Opt. Commun. Netw., vol. 7, no. 2, pp. A174–A182, Feb. 2015.

[4] C. Filsfils et al., "Segment routing architecture," draft-filsfils-springsegment-routing, 2015.

[5] https://www.opennetworking.org/sdn-resources/openflow

[6] S. Peng et al. "A Novel SDN enabled Hybrid Optical Packet/Circuit Switched Data Centre Network: the LIGHTNESS approach", to be presented in EuCNC 2014 conference, Bologna, Italy, June 23-26, 2014

[7] B. Guo et al. "SDN-enabled Programmable Optical Packet/Circuit Switched Intra Data Centre Network ", OFC 2015, Los Angeles, CA, US, March 22-26, 2015.

[8] A. Singh et al. "Jupiter rising: A decade of clos topologies and centralized control in Google's datacenter network," ACM SIGCOMM Computer Communication Review, Vol. 45, pp. 183-197, September 2015

[9] Sridhar K. N. Rao, "SDN and its use-cases- NV AND NFV: A State-of-the-Art Survey," http://www.nectechnologies.in/en_TI/pdf/NTI_whitepaper_SDN_NFV.pdf

[10] Steven S. W. Lee, Kuang-Yi Li, Kwan-Yee Chan, Guan-Hao Lai, and Yao-Chuan Chung, "Path Layout Planning and Software based Fast Failure Detection in Survivable OpenFlow Networks," DRCN, 2014.

[11] Steven S. W. Lee, Kuang-Yi Li, Kwan-Yee Chan, Guan-Hao Lai, and Yao-Chuan Chung, "Software-based Fast Failure Recovery for Resilient OpenFlow Networks," Reliable Networks Design and Modeling (RNDM), 2015.

[12] C. Cascone, L. Pollini, D.Sanvito, A. Capone, B. Sanso, "SPIDER: Fault Resilient SDN Pipeline with Recovery Delay Guarantees," IEEE NetSoft Conference and Workshops (NetSoft), 2016.

[13] M. Xia, M. Shirazipour, H. Mahkonen, R. Manghirmalani, A. Takacs "Resource Optimization for Service Chain Monitoring in Software-Defined Networks," Fourth European Workshop on Software Defined Network, 2015.

[14] H. Mahkonen, R. Manghirmalani, M. Shirazipour, M. Xia, A. Takacs, "Elastic Network Monitoring With Virtual Probes," IEEE Conference on Network Function Virtualization and Software Defined Networks 2015.

[15] M. Shirazipour, H. Mahkonen, V. S. Vega, "A Monitoring Framework at Layer4-7 Granularity Using Network Service Headers," IEEE Conference on Network Function Virtualization and Software Defined Network (NFV-SDN) 2015.

[16] B. Patil, S. Kinger , V. K. Pathak , "Probe Station Placement Algorithm for Probe Set Reduction in Network Fault Detection and Localization", 19th IEEE International Conference on Networks (ICON), 2013.

[17] M. Natu, A. S. Sethi, "Efficient probing techniques for fault diagnosis", Second International Conference on Internet Monitoring and Protection, IEEE, 2007.

[18] S. Kandula, S. Sengupta, A. Greenberg, P. Patel, and R. Chaiken, "The nature of data center traffic: measurements and analysis", 9th ACM SIGCOMM conference on Internet measurement conference, USA, 2009.

[19] T. Benson, A. Akella, and D. A. Maltz, Network traffic characteristics of data centers in the wild, ACM SIGCOMM on Internet Measurement Conference, 2010.

[20] Sachin Sharma, Dimitri Staessens, Didier Colle, Mario Pickavet, and Piet Demeester, "Enabling Fast Failure Recovery in OpenFlow Networks," DRCN, 2011.

[21] Dimitri Staessens, Sachin Sharma, Didier Colle, Mario Pickavet, and Piet Demeester, "Software Defined Networking: Meeting Carrier Grade Requirement," IEEE LANMAN, 2011.

[22] J. Kempf, E. Bellagamba, A. Kern, D. Jocha, A. Takacs, and P. Skoldstrom, "Scalable Fault Management for OpenFlow," IEEE ICC, 2012.

[23] A. Sgambelluri, A. Giorgetti, F. Cugini, F. Paolucci, P. Castoldi, "OpenFlow-based Segment Protection in Ethernet Networks," IEEE/OSA Journal of Optical Communications and Networking, vol.5, no.9, pp.1066-1075, Sept. 2013.

[24] Y. Zhao, P. Zhang, Y. Jin, "Netography: Troubleshoot Your Network with Packet Behavior in SDN," IEEE/IFIP Network Operations and Management Symposium (NOMS), 2016.

[25] http://openstate-sdn.org/.

[26] http://openvswitch.org/

[27] https://osrg.github.io/ryu/