Computer Networks 133 (2018) 195–211

Contents lists available at ScienceDirect

Computer Networks

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

# On reliability improvement of Software-Defined Networks

Shadi Moazzeni[a], Mohammad Reza Khayyambashi[a,*], Naser Movahhedinia[a], Franco Callegati[b]

[a] Department of Computer Architecture, Faculty of Computer Engineering, University of Isfahan, Isfahan, Iran
[b] Department of Computer Science and Engineering, University of Bologna, via Venezia 52, Cesena, FC 47521, Italy

A R T I C L E I N F O

*Article history:*
Received 26 March 2017
Revised 6 November 2017
Accepted 17 January 2018

*Keywords:*
Software-Defined Networks
Distributed controllers
Reliability
Failure detection
Fast failure recovery
Coordinator controller

A B S T R A C T

In Software-Defined Networks (SDNs) the role of the centralized controller is crucial, and thus it becomes a single point of failure. In this work, a distributed controller architecture is explored as a possible solution to improve fault tolerance. A network partitioning strategy, with small subnetworks, each with its own Master controller, is combined with the use of Slave controllers for recovery aims. A novel formula is proposed to calculate the reliability rate of each subnetwork, based on the load and considering the number and degree of the nodes as well as the loss rate of the links. The reliability rates are shared among the controllers through a newly-designed East/West bound interface, to select the coordinator for the whole network. This proposed method is called "Reliable Distributed SDN (RDSDN)." In RDSDN, the failure of controllers is detected by the coordinator that may undertake a fast recovery scheme to replace them. The numerical results prove performance improvement achievable with the adoption of the RDSDN and show that this approach performs better regarding failure recovery compared to methods used in related research.



## 1. Introduction and motivation

Software-Defined Networking (SDN) has recently emerged as a novel paradigm to overcome the challenges related to the control plane of modern communication networks [1,2]. The brain of the control plane is the so-called SDN controller, which typically talks with network devices through a Southbound Interface (SBI) such as the OpenFlow protocol [3]. The control plane exposes some features and APIs through the Northbound Interface (NBI) to network operators to design various management applications exploiting, for instance, a set of REST APIs [4,5]. The centralized control plane approach of SDN promises controllable networks but raises a reliability issue since the SDN controller may turn into a centralized point of failure. This is a known issue, and several countermeasures have been proposed. We have reviewed these works in Section 2.

In this article the goal is to consider the data plane and control plane reliability as a combined issue, proposing a solution that combines network partitioning, controllers' coordination, and data plane reliability characteristics to enhance the overall network resilience.

To reduce the effect of the data plane or controller failures, it is assumed that a whole network domain can be partitioned into subnetworks. Each subnetwork is controlled by a *Master* controller and has one or more controllers of the other subnetworks as *Slave* controllers. Each subnetwork's Master controller calculates the reliability rate by exploiting the newly proposed formula. The reliability rates are shared periodically among controllers using edge switches through a newly designed East/West bound interface. There may be backup control routes in addition to the main routes to improve fault coverage. The controller which has the best reliability rate would be selected as the *coordinator* who checks the status of the other controllers, periodically. This newly proposed method is called "Reliable Distributed SDN (RDSDN)" which aims to improve the reliability of SDNs with distributed controllers. Through the detection phase, the coordinator detects any non-active controller and will decide which other controller is more appropriate to take over the subnetwork according to the cached reliability rates and then will trigger the fast recovery scheme until the failed controller is repaired. Therefore, the created inertia is attenuated. If the coordinator crashes or a better controller exists, a new one will be chosen by election.

The paper is organized as follows: A review of the most important issues in SDN reliability and the related studies are presented in Section 2. The main contribution containing the state-of-the-art method for calculating the reliability rate and describing RDSDN is in Section 3. The pilot implementation of our work, including

* Corresponding author.
*E-mail addresses:* moazzeni@eng.ui.ac.ir (S. Moazzeni), m.r.khayyambashi@comp.ui.ac.ir (M.R. Khayyambashi), naserm@eng.ui.ac.ir (N. Movahhedinia), franco.callegati@unibo.it (F. Callegati).
*URL:* http://eng.ui.ac.ir/~m.r.khayyambashi (M.R. Khayyambashi)

https://doi.org/10.1016/j.comnet.2018.01.023
1389-1286/

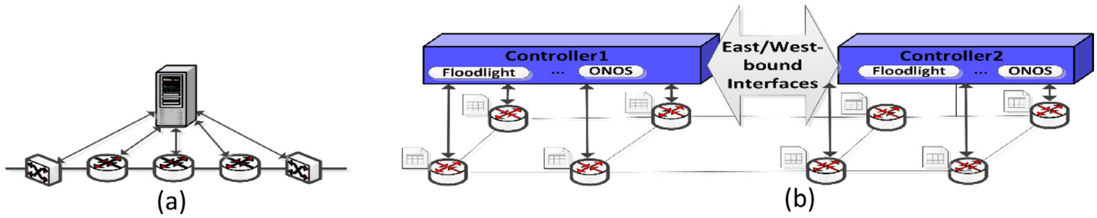


**Fig. 1.** SDNs with (a) central controller (b) distributed controllers.

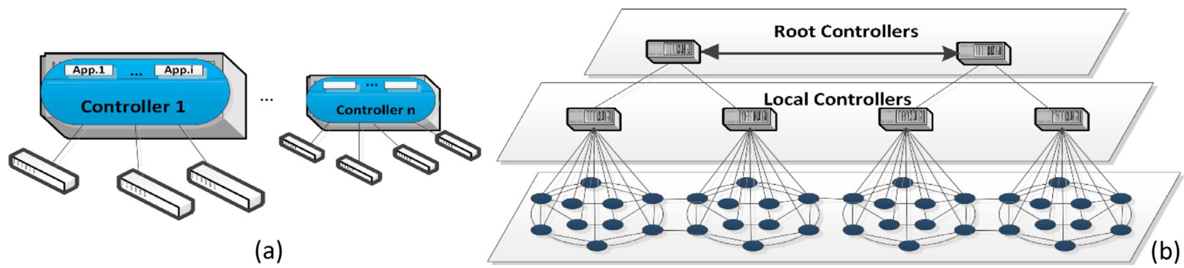


**Fig. 2.** A view of distributed controllers (a) fully-distributed (flat) (b) hierarchical.

failure detection and recovery schemes, is presented in Section 4. The numerical results are presented in Section 5, and finally, in Section 6 a brief conclusion is given.

## 2. Reliability issues in SDNs

An SDN implementation may be based on a centralized (Fig. 1a) or distributed (Fig. 1b) controller architecture. The former solution is simpler and easier to manage but intrinsically unreliable. Moreover, the effectiveness of the controller may be impaired by the propagation delay when the distance between switches and controller is large [6]. The latter is more complex. It requires that, for consistency reasons, the controllers talk to each other through the so-called East/West bound interfaces but is also known to be effective in improving fault tolerance and reliability [6–9]. Briefly, we can say that a distributed controller architecture is preferable when reliability is the issue.

Nonetheless the control plane topology and network elements distribution level still present a number of alternatives with different characteristics. It is evident that the distribution of the network elements among the controllers could lower the effect of inevitable physical failures on the controlling networks and improve the overall reliability. As shown in Fig. 2a and b, in control plane setup, controller arrangement is either fully-distributed (flat arrangement) or hierarchical (vertical arrangement). The fully-distributed architecture may require a considerable amount of synchronization overhead to integrate the controllers while the hierarchical architecture may not tolerate all the errors and failures if the Master is the point of failure. In a hierarchical arrangement, only the root controller owns and manages the global network state. Conversely, in a flat arrangement, each SDN controller has the global network-wide state [10].

### 2.1. Control plane topology

#### 2.1.1. Fully distributed (flat) controllers

Onix [11], SmartLight [12], DISCO [13], ElastiCon [14], ICONA [15], and cluster-based distributed controllers [16] are examples of fully-distributed controllers. The majority of these controllers have a consistent view. In Onix, the topology information mostly in static mode is distributed among all controllers under replicated databases while in dynamic mode is distributed in a hash table with weak consistency. In a SmartLight controller, the network is controlled by a central controller, and backup controllers are used as active replicas to create constant views of the network to ensure fault tolerance. However, the applicants' requests need to be forwarded to all replications which is time-consuming.

DISCO distributed controllers rely on agents to supply an end-to-end service. The message that is sent to a region is sent to other regions as well. Therefore, DISCO does not emphasize a solution to enable a controller to consider links with higher performance or stopping the transfer of excessive messages among different regions.

Given that there is a high probability of traffic and load change on a controller in heterogeneous networks, loads in ElastiCon are considered to allow a switch to migrate from a crowded controller to another less crowded one. The network in the ICONA application – which is used as an upstream application in ONOS open source OS [17]– is categorized into several clusters where each cluster consists of a head controller and some backups. Making connections among controllers to quantify traffic using ONOS is considered in [18]. Even though the fault tolerance is high due to using backup replicas, there may exist a considerable synchronization overhead due to the generation of a strong consistency which may decrease the overall performance. In the proposed RDSDN method, since each SDN controller owns the global network-wide state, it is placed in the flat layer. On the other hand, because of using the Master/Slave feature, when the coordinator detects a master controller failure, the most reliable controller between the preconfigured slave controllers will be chosen, and the failed switches will be assigned to it. So, the synchronization overhead is respectively lower.

#### 2.1.2. Hierarchical controllers

Kandoo [9], Improved Kandoo [19], DSDN [20] and FlowBroker [21] are examples of two-level hierarchical controllers. ORION [22], ANT [23], and multi-level controllers [24] are examples of multi-level hierarchical controllers, and heterogeneous multi-level hierarchical controllers [25] is an example of the clustered hierarchical controllers.

In most hierarchical controllers, two layers, including the root controller and local controllers are taken into account. In other solutions like those introduced in [8,24], the topology of the control plane is examined out-of-band with three layers through identical links. In the article introducing heterogeneous multi-level hierarchical controllers, controllers are clustered and to reduce the delay