Title: METHOD AND SYSTEM FOR BALANCING LOAD IN A SDN NETWORK

Abstract: The present invention relates to a method for balancing load in a cluster (1) of SDN controllers (2). Initially, for an application (3) and/or device (4) intending to connect to the cluster (1), a primary SDN controller (2) is defined by a CMS. The application (3) and/or device (4) are then connected to the primary SDN controller (2). Then, the primary SDN controller (2) determines a best SDN controller (2) for the application (3) and/or device (4) based on load data (6) of the cluster (1) and/or based on other characteristics like utilization (CPU, memory, I/O etc.). Then, the primary SDN controller (2) instructs the application (3) and/or device (4) to connect to the best SDN controller (2), causing the application (3) and/or device (4) to finally connect to the best SDN controller (2). To perform the method, the cluster (1) preferably uses a coordination cluster (5) for storing and locking the load data (6) into a predetermined path of the coordination cluster (5). The present invention also relates to a system comprising a cluster (1) of SDN, controllers (2) configured to carry out said method.
Method and system for balancing load in a SDN network

TECHNICAL FIELD

The present invention relates to a method and system for balancing load in a cluster of software defined network (SDN) controllers. In particular, the method and system of the present invention balances a load of applications and/or devices intending to connect to the cluster of SDN controllers.

BACKGROUND

In the state of the art, SDN is an emerging network technology addressing customization and optimization concerns. In SDN the data plane is decoupled from the control plane, and thus modern communication networks may be simplified. The control plane functions are normally implemented in one or more SDN controllers. SDN controllers may be grouped into clusters. Typically a cluster of distributed SDN controllers has to simultaneously handle a large number of connected applications and devices in the SDN. For example, one or more devices, such as a virtual switch (vSwitch), and one or more applications, such as deep packet inspection (DPI), firewall (FW), web application firewalls (WAF) and more, may be connected to a cluster of distributed SDN controllers.

A cluster of SDN controllers is preferably designed such that the load on the individual SDN controllers in the cluster is balanced. In other words, each SDN controller should serve the same number of devices and/or applications, irrespectively of the actual total load of the cluster.

To this end, the state of the art traditionally implements a load balancer (LB) between the application layer and a Northbound Interface (NBI) of the SDN controllers in the cluster. Furthermore, another LB is traditionally implemented between the device layer and a Southbound Interface (SBI) of the SDN controllers in the cluster. However, each of these LBs is a bottleneck, and actually each LB is thus a potential single point of failure (SPOF). This means that in case of a failure of a LB, the whole
cluster becomes unavailable. Such a SPOF is in stark contrast to a main concern of a cluster of SDN controllers, namely that the cluster provides high availability. High availability is a system design approach and an associated service implementation, which intends to ensure a certain level of operational performance. In particular, the term availability refers in the present invention to the ability of a user and/or application and/or device to obtain a service via a cluster of SDN controllers or to access the cluster. If the cluster is inaccessible, or if a service may not be obtained via the cluster, the cluster is not available to the user and/or application and/or device. For high availability, the cluster, and specifically the services and applications running on the SDN controllers in the cluster, are preferably available at all times.

SUMMARY

In view of the above problems and disadvantages, the present invention intends to improve the state of the art. In particular, an object of the present invention is to balance a load between applications and/or devices and SDN controllers contained in a cluster of SDN controllers, without creating a SPOF, which endangers the whole cluster to become unavailable.

The above-mentioned object is achieved by the solution provided in the enclosed independent claims. Advantageous implementations are defined in the respective dependent claims. In particular, the core of the invention is to mitigate the SPOFs of traditional LBs by a distributed load balancing mechanism, which is implemented in a software based manner on each SDN controller of a cluster using a coordination service for synchronizing the loads between the individual SDN controllers.

A first aspect of the present invention provides a method for balancing load in a cluster of SDN controllers, the method comprising defining for an application and/or device, intending to connect to the cluster, a primary SDN controller, connecting, by the application and/or device, to the primary SDN controller, determining, by the primary SDN controller, a best SDN controller based on load data of the cluster, instructing, by the primary SDN controller, the application and/or device to connect to the best SDN controller, connecting, by the application and/or device, to the best SDN controller.
In the method of the present invention, each application or device connecting to the cluster connects particularly to the primary SDN controller instead of connecting to a common LB as in the state of the art. The LB is thus eliminated as SPOF. Further, by determining a best SDN controller, the method of the present invention is able to automatically balance the load among the SDN controllers in the cluster. Each new application and/or device is preferably instructed to connect to the best SDN controller in view of the load distribution in the cluster, i.e. preferably to the SDN controller currently handling the least amount of load, for example, the lowest number of applications and/or devices.

In a first implementation form of the method of the first aspect, the method further comprises storing load data of the cluster using a predetermined path of a coordination cluster, wherein the step of determining the best SDN controller comprises locking, by the primary SDN controller, said predetermined path of the coordination cluster, reading, by the primary SDN controller, the load data of the cluster stored in the predetermined path of the coordination cluster, calculating, by the primary SDN controller, the best SDN controller, updating, by the primary SDN controller, the estimated load data of the cluster in the coordination cluster according to a load estimation based on the determined best SDN controller, unlocking, by the primary SDN controller, the predetermined path of the coordination cluster, and reconfiguring, by the SDN controller, the application and/or device by the calculated best SDN controller.

The coordination cluster is preferably based on Apache Zookeeper and is used for a synchronizing and locking mechanism. Thereby, it is guaranteed that each SDN controller always operates based on the newest and most relevant load data. The coordination cluster provides the load data with high availability to the SDN controllers by using several nodes, wherein read/write can be carried out over any node. The load balancing in the cluster of SDN controllers can be carried out more efficiently and more accurate.

After the step of unlocking, the primary SDN controller instructs the application and/or device to connect to the best SDN controller. Thereby, the primary SDN
controller reconfigures the device, preferably by using OF-Config, by the best SDN controller the primary SDN controller found. Then the application and/or device connect to the best SDN controller. Finally, the best SDN controller updates the real load data in the coordination cluster.

In a second implementation form of the method according to the first implementation form of the first aspect, the method further comprises updating, by the best SDN controller, the load data of the cluster in the coordination cluster, when the application and/or device connects to the best SDN controller.

In a third implementation form of the method according to the first aspect as such or according to any of the previous implementation forms of the first aspect, the coordination cluster is implemented by Apache Zookeeper.

Apache Zookeeper is a popular and scalable coordination cluster, which promises coordination and consistency between multiple of its nodes. The SDN controllers can obtain the load data from any node of Apache Zookeeper. Thus, high availability of the load data can be guaranteed.

Preferably, the cluster of SDN controllers includes 2 to 20 SDN controllers. The coordination controller preferably includes 3 to 5 nodes.

In a fifth implementation form of the method according to the first aspect as such or according to any of the previous implementation forms of the first aspect, the method further comprises defining for the application and/or device, intending to connect to the cluster, a secondary SDN controller, and substituting the primary SDN controller for the secondary SDN controller in all relevant method steps, in case of a connection failure to the primary SDN controller.
By defining the second SDN controller for each application and/or device, the method is robust against a failure of the primary SDN controller. In case a primary SDN controller fails, or a connection to or from a primary SDN controller fails, the related secondary SDN controller immediately takes over all functions and steps. SPOFs of the cluster of SDN controllers are thereby eliminated. Thus, the cluster provides high availability. It is also possible to define more than two SDN controllers per application and/or device. For example, it is even possible to provide each application and/or device with a priority order of all SDN controllers in the cluster. Thus, each application and/or device will always be able to connect to an initial SDN controller, which can then determine the best SDN controller.

In a sixth implementation form of the method according to the first aspect as such or according to any of the previous implementation forms of the first aspect, the step of defining the primary SDN controller is carried out by one or more cloud management system, CMS.

The one or more CMS is, for example, Open Stack, Cloud Stack or the like. The CMS can use a round robin algorithm per cluster to define the primary and the secondary SDN controllers for each device and/or application. Thereby, initial load balancing is guaranteed during handshake. It is possible to use multiple CMS for redundancy, in order to achieve high availability.

In a seventh implementation form of the method according to the first aspect as such or according to any of the previous implementation forms of the first aspect the load data of the cluster comprises the number of applications and/or devices connected to each SDN controller of the cluster and/or comprises memory utilization, disk space, CPU utilization and/or I/O rates of each SDN controller.

The load data can alternatively or additionally be calculated based on a weighted logistic regression formula based on features like memory, CPU, I/O or the like. In other words, the memory utilization of each SDN controllers, the processing power (CPU utilization) of each SDN controller, or the I/O rate of each SDN controller in the cluster can be taken into account.
In an eighth implementation form of the method according to the first aspect as such or according to any of the previous implementation forms of the first aspect, the step of determining the best SDN controller comprises using a weighted round robin algorithm.

In a ninth implementation form of the method according to the first aspect as such or according to any of the previous implementation forms of the first aspect, if a plurality of applications and/or devices intend to connect to the cluster at the same time, the method comprises defining a plurality of primary SDN controllers, a different primary SDN controller for each application and/or device, connecting, by each application and/or device, to its primary SDN controller, determining, by each of the plurality of primary SDN controllers, one primary SDN controller after the other according to an order, a best SDN controller for the application and/or device connected to it based on load data of the cluster, instructing, by each of the plurality of primary SDN controller, the application and/or device connected to it to connect to the best SDN controller determined for it, connecting, by each application and/or device, to its best SDN controller.

The plurality of primary SDN controllers, the different primary SDN controller for each application and/or device can be defined by using a round robin algorithm.

The method can thereby ensure that for each new application and/or device the actual best SDN controller may be found. Basically, each SDN controller can read/write from any node of the coordination cluster. However, if locked it is ensured that the load data in the predetermined path of the coordination cluster is only by accessed by one SDN controller at a time.

Each application is associated with one primary SDN controller, wherein the SDN controllers of different applications and/or devices differ from each other. Preferably, the CMS calculates, preferably based on a round robin algorithm, the primary SDN controller of each application and/or device. Preferably, the CMS carries out proactive load balancing when defining the plurality of SDN controllers for the plurality of applications and/or devices.
In a tenth implementation form of the method according to the ninth implementation form of the first aspect, determining the best SDN controllers comprises waiting, by each primary SDN controller, for its turn in the order, preferably waiting in a stand-by mode.

Therefore, the method of the present invention can be carried out in an efficient and energy-saving manner.

In an eleventh implementation form of the method according to the first aspect as such or according to the ninth or tenth implementation forms of the first aspect, the method further comprises defining a plurality of SDN controllers, a different SDN controller for each application and/or device, and substituting each primary SDN controller exhibiting a failure for its corresponding secondary SDN controller in all relevant method steps.

The substitution of the primary SDN controller to the secondary SDN controller ensures high availability of the cluster.

In a second aspect the present invention provides a system comprising a cluster of software defined network, SDN, controllers, the system being configured to define for an application and/or device intending to connect to the cluster a primary SDN controller, connect the application and/or device to the primary SDN controller, determine, by the primary SDN controller, a best SDN controller based on load data of the cluster, instruct, by the primary SDN controller, the application and/or device to connect to the best SDN controller, connect the application and/or device to the best SDN controller.

In a first implementation form of the system of the second aspect, the system further comprises a coordination cluster for storing load data of the cluster using a predetermined path, wherein for determining the best SDN controller the system is configured to lock, by the primary SDN controller, said predetermined path of the coordination cluster, read, by the primary SDN controller, the load data of the cluster stored in the predetermined path of the coordination cluster, calculate, by the primary SDN controller, the best SDN controller, update, by the primary SDN controller, the
load data of the cluster in the coordination cluster according to a load estimation based on the determined best SDN controller, unlock, by the primary SDN controller, the predetermined path of the coordination cluster, and reconfigure, by the SDN controller, the application and/or device by the calculated best SDN controller.

In a second implementation form of the system of the first implementation form of the second aspect, the system is further configured to update, by the best SDN controller, the load data of the cluster in the coordination cluster, when the application and/or device is connected to the best SDN controller.

The system of the present invention achieves the same advantages as described above for the method. The system of the present invention may be further configured to carry out the method steps according to all implementation forms described above.

BRIEF DESCRIPTION OF DRAWINGS

The above aspects and implementation forms of the present invention will be explained in the following description of specific embodiments in relation to the enclosed drawings, in which

Fig. 1 shows a basic flow of a method according to an embodiment of the present invention.

Figs. 2 to 9 show a specific example of the basic flow of the method shown in Fig. 1, which is carried out in a system according to an embodiment of the present invention.

DETAILED DESCRIPTION OF EMBODIMENTS

Fig. 1 shows a sequence diagram, which demonstrates a basic flow of an embodiment of the method of the present invention. The method implements a distributed load balancing mechanism for a cluster 1 of SDN controllers 2. The method is particularly able to handle huge traffic and requests in both directions to and from the cluster 1 of SDN controllers 2.
The method is preferably carried out in an SDN environment, which preferably includes the cluster 1 of SDN controllers 2, at least one SDN application 3, like DPI, WAF etc., at least one devices 4, like a vSwitch, and a coordination cluster 5.

In an initial step of the method, for each application 3 and/or device 4 intending to connect to the cluster 1 of SDN controllers 2, a primary SDN controller 2 is defined for and configured in the application 3 and/or device 4. As shown in Fig. 1, preferably a cloud management system (CMS) 8 is responsible for defining and configuring the primary SDN controller 2 for each application 3 and/or device 4. Preferably, also a secondary SDN controller 2 is defined for and configured in each application 3 and/or device 4 by the same CMS 8. Each secondary SDN controller 2 functions as a backup for its related primary SDN controller 2 (i.e. the primary SDN controller 2 determined for and configured in the same application 3 and/or device 4), in order to replace the related primary SDN controller 2, for example, in case said related primary SDN 2 controller fails or in case a connection to the related primary SDN controller 2 fails.

Preferably, the CMS 8 uses a round robin algorithm to define for each application 3 and/or device 4 the primary SDN controller 2 and preferably the secondary SDN controller 2. By using a round robin algorithm, an initial load balancing is guaranteed during the handshake of an application 3 and/or device 4 and the cluster 1 can be provided. It is also possible to use more than one CMS 8 for reasons of redundancy, and thus in order to guarantee a high availability of the cluster 1. In the following, the further method steps are described in relation to a primary SDN controller 2 defined per application 3 and/or device 4. However, the method automatically switches to one or more secondary SDN controllers 2 at any time or any step in the method, if the connection to one or more related primary SDN controllers 2 fails.

In a further method step, an application 3 and/or device 4, for which the CMS 8 has defined and configured the primary SDN controller 2 and optionally the secondary SDN controller 2, connects to its primary SDN controller 2. This is in contrast to the state of the art, where an application 3 and/or device 4 have to connect to a common LB, which presents a SPOF. After the application 3 and/or device 4 has connected to its primary SDN controller 2, a best SDN controller 2 is determined for the application 3 and/or device 4 taking into account the load data 6 of the cluster 1. The coordination cluster 5 is preferably implemented by Apache Zookeeper, and comprises a plurality...
of nodes for communicating with the cluster 1. Apache Zookeeper promises good
coordination capabilities and reliable consistency between its individual nodes. The
method according to the embodiment shown in Fig. 1, stores the load data 6 of the
cluster 1 in a predetermined path of the coordination cluster 5. The predetermined path
can, for example, be identified by "clusterID/SDNcontroller ID/applicationsID/devices".
The load data 6 can, for example, include the number of applications 3 and/or devices
4, which are connected to each individual SDN controller 2 of the cluster 1. The load
data 6 can, however, also be based on different characteristics of the SDN controllers
2, for example, their memory and CPU utilization, processing capabilities, and/or I/O
rate. Preferably, a weighted formula may be used to calculate the load data 6 based on
these different characteristics.

In a further step, after the application 3 and/or device 4 has connected to its primary
SDN controller 2 (or its secondary SDN controller 2 in case of a connection failure of
the primary SDN controller 2), said primary SDN controller 2 locks the load data 6 on
the predetermined path of the coordination cluster 5. Thus, no other SDN controller 2
of the cluster 1 can now access the locked load data 6 on the predetermined path. Due
to the coordination capabilities of Apache Zookeeper, the load data 6 can be read from
any one of its nodes, and thus provides high availability. The primary SDN controller
2 not only locks the load data 6 on the predetermined path of the coordination cluster
5, but also reads in a further step the load data 6 stored in the predetermined path.
Based on the obtained load data 6, the primary SDN controller 2 can then determine a
best SDN controller 2 for the application 3 and/or device 4 currently intending to
connect to the cluster 1. For example, the primary SDN controller 2 defines and
unutilized SDN controller 2, which the application 3 and/or device 4 may connect to,
as the best SDN controller 2. Preferably, a basic weighted round robin algorithm is
used to find the best SDN controller 2.

When a best SDN controller 2 has been found by the primary SDN controller 2, the
primary SDN controller 2 updates in a further step of the method the estimated load
data 6 into the predetermined path of the coordination cluster 5, i.e. the load data 6
assuming that the application and/or device will connect to the determined best SDN
controller 2, and unlocks the predetermined path. Furthermore, the primary SDN
controller 2 sends an instruction message to the application 3 and/or device 4
(preferably via OF-Config protocol), which intends to connect to the cluster 1, in order to instruct it to connect to the determined best SDN controller 2.

Based on the instruction request from its primary SDN controller 2, the application 3 and/or device 4 updates in a further method step its configuration, i.e. it is re-configured, in order to substitute the preconfigured primary SDN controller 2 by the best SDN controller 2. Then the application 3 and/or device 4 connect to its best SDN controller 2. When the application 3 and/or device 4 has connected to its best SDN controller 2, said best SDN controller 2 locks the load data 6 on the predetermined path of the coordination cluster 5 in a further step, and updates the real new load data 6 onto the predetermined path. Finally, the best SDN controller 2 unlocks the predetermined path. Within the coordination controller 5, the information stored in the predetermined path may be synched between a leader node and a follower node for reasons of data integrity and security.

Figures 2 to 9 show an example for how an embodiment of the method of the present invention is performed in an embodiment of a system of the present invention. In the figs. 2 to 9 the system comprises a cluster 1 of SDN controllers 2, and preferably comprises a coordination cluster 5 for storing load data 6 of the cluster 1 using a predetermined path. The cluster 1, i.e. particularly the SDN controllers 2 of the cluster 1, can connect and communicate with the coordination cluster 5 via a plurality of nodes 7 of the coordination cluster 5.

To the cluster 1 of SDN controllers 2, a plurality of applications 3 and/or devices 4 may be connected or may intend to connect. In the example shown in the figs. 2 to 9, three SDN controllers 2 (in the following distinguished by 2a, 2b and 2c) are included in the cluster 1. Additionally, the coordination cluster 5 is realized by Apache Zookeeper, which comprises three Zookeeper nodes 7. In the predetermined path the load data 6, in this case the number of applications 3 and devices 4 in the cluster 1 for each SDN controller 2 is stored. In the example according to figs. 2 to 9, two applications 3 are connected to each SDN controller 2a, 2b and 2c. Additionally, the SDN controller 2a and the SDN controller 2c each handle nine devices 4, while the SDN controller 2b handles eight devices 4.
Fig. 3 now shows a case where two new devices 4a and 4b, for example both vSwitches, want to connect to the cluster 1 of the SDN controllers 2. The new devices 4a and 4b are both already configured, preferably by the CMS 8 described in relation to Fig. 1, with a primary SDN controller 2 and optionally with a secondary SDN controller 2 for each device 4a and 4b. In the present example, the device 4a shown in Fig. 3 is configured with SDN controller 2a as its primary SDN controller, and the right device 4b is configured with SDN controller 2c as its primary SDN controller. Furthermore, both devices 4a and 4b may be configured with a secondary SDN controller 2 for redundancy in case of a failure of a primary SDN controller 2. For example, the device 4a is configured with SDN controller 2b as its secondary SDN controller, and the device 4b is configured with SDN controller 2c as its secondary SDN controller. In Fig. 3 the two devices 4a and 4b intending to connect to the cluster 1 connect simultaneously to their pre-defined primary SDN controllers 2a and 2c, respectively, and send a request for being instructed with a best SDN controller 2.

In Fig. 4 the lock mechanism of the coordination cluster 5 is employed. For example, first the SDN controller 2c locks the predetermined path on the coordination cluster 5, and obtains the load data 6 stored therein. Then, the SDN controller 2c calculates the best SDN controller 2 to use for its device 4b. Thereby, it preferably uses a weighted round robin algorithm. While the SDN controller 2c determines the best SDN controller 2, the SDN controller 2a cannot read data from the coordination cluster 5 and is preferably in a waiting state, more preferably in a stand-by state.

In Fig. 5 the SDN controller 2c determines that the best SDN controller 2 for the device 4b is in the SDN controller 2b, because it has currently only eight devices 4 connected to it, while the other SDN controllers 2a and 2c handle each nine connected devices 4. Accordingly, the SDN controller 2c updates the estimated load data 6 in the predetermined path of the coordination cluster 5, i.e. a load estimation based on the assumption that the device 4b will connect to the SDN controller 2b, and sends a response to the device 4b, in order to instruct it to use the SDN controller 2b as best SDN controller to connect to. Then, the SDN controller 2c unlocks the data load 6 on the predetermined path of the coordination cluster 5.
In Fig. 6 now the SDN controller 2a can read the load data 6 from the coordination cluster 5, and therefore locks the predetermined path. Similar as the SDN controller 2c before, it defines the best SDN controller 2 for the device 4a, preferably again by using a weighted round robin algorithm. In Fig. 6 the best SDN controller 2 for the device 4a is the SDN controller 2a, which is also the primary SDN controller of the device 4a.

As explained before, in Fig. 7 the SDN controller 2a updates the estimated load data 6 on the predetermined path of the coordination cluster 5, i.e. the load estimated assuming that the device 4a will stay connected with the SDN controller 2a, and instructs the device 4a to use the SDN controller 2a as its best SDN controller.

In Fig. 8 the device 4a reconnects to SDN controller 2a (actually it is already connected so it preferably just changes the state flag from "initial" to "real"). Then each of the best SDN controllers 2a and 2b update the real load data on the predetermined path of the coordination cluster 5.

Fig. 9 shows a final state of the system, after the two devices 4a and 4b have connected to the cluster 1. The cluster 1 is load balanced. The SDN controller 2a now handles ten devices 4, while the SDN controller 2b and the SDN controller 2c handle each nine devices 4. Thus, the figs. 2 to 9 demonstrate a method and a system of an embodiment of the present invention for a distributed load balancing mechanism. In particular, the load in a cluster 1 of SDN controllers 2 is balanced, without creating a SPOF or availability bottleneck. Thus, a SDN with high availability can be supported.

It is noted that the flow described within the figs. 2 to 9 for the devices 4a and 4b, which intended to connect to the cluster 1 of SDN controllers 2 via a SBI works likewise for balancing the load between applications 3 intending to connect via the NBI to the cluster 1 of SDN controllers 2.

In summary the method and the system of the present invention achieve load balancing mechanism for a cluster 1 of SDN controllers 2. Advantageously, the load balancing is implemented such that no SPOF is created. Since no SPOF is created, the SDN performance is increased, and high availability can be guaranteed at all times.
The invention has been described in conjunction with various embodiments herein. However, other variations to the disclosed embodiments can be understood and effected by those skilled in the art in practicing the claimed invention, from a study of the drawings, the disclosure, and the appended claims. In the claims, the word "comprising" does not exclude other elements or steps, and the indefinite article "a" or "an" does not exclude a plurality. A single processor or other unit may fulfill the functions of several items recited in the claims. The mere fact that certain measures are recited in mutually different dependent claims does not indicate that a combination of these measured cannot be used to advantage. A computer program may be stored/distributed on a suitable medium, such as an optical storage medium or a solid-state medium supplied together with or as part of other hardware, but may also be distributed in other forms, such as via the Internet or other wired or wireless telecommunication systems.
Claims

1. Method for balancing load in a cluster (1) of software defined network, SDN, controllers (2), the method comprising
   defining for an application (3) and/or device (4), intending to connect to the cluster (1), a primary SDN controller (2),
   connecting, by the application (3) and/or device (4), to the primary SDN controller (2),
   determining, by the primary SDN controller (2), a best SDN controller (2) based on load data (6) of the cluster (1),
   instructing, by the primary SDN controller (2), the application (3) and/or device (4) to connect to the best SDN controller (2),
   connecting, by the application (3) and/or device (4), to the best SDN controller (2).

2. Method according to claim 1, further comprising
   storing load data (6) of the cluster (1) using a predetermined path of a coordination cluster (5),
   wherein the step of determining the best SDN controller (2) comprises
   locking, by the primary SDN controller (2), said predetermined path of the coordination cluster (5),
   reading, by the primary SDN controller (2), the load data (6) of the cluster (1) stored in the predetermined path of the coordination cluster (5),
   calculating, by the primary SDN controller (2), the best SDN controller (2),
   updating, by the primary SDN controller (2), the estimated load data of the cluster (1) in the coordination cluster (5) according to a load estimation based on the determined best SDN controller (2),
   unlocking, by the primary SDN controller (2), the predetermined path of the coordination cluster (5), and
   reconfiguring, by the SDN controller, the application and/or device by the calculated best SDN controller (2).

3. Method according to claim 2, further comprising
updating, by the best SDN controller (2), the load data (6) of the cluster (1) in
the coordination cluster (5), when the application (3) and/or device (4) connects to the
best SDN controller (2).

5 4. Method according to one of the claims 1 to 3, wherein the coordination
cluster (5) is provided with a plurality of nodes (7) for the SDN controller (2) of the
cluster (1).

5 5. Method according to one of the claims 1 to 4, further comprising
defining for the application (3) and/or device (4), intending to connect to the
cluster, (1) a secondary SDN controller (2), and
substituting the primary SDN controller (2) for the secondary SDN
controller (2) in all relevant method steps, in case of a connection failure to the
primary SDN controller (2).

6 6. Method according to one of the claims 1 to 5, wherein the step of defining the
primary SDN controller (2) is carried out by one or more cloud management system,
CMS (8).

7 7. Method according to one of the claims 1 to 6, wherein the load data (6) of the
cluster (1) comprises the number of applications (3) and/or devices (4) connected to
each SDN controller (2) of the cluster (1) and/or comprises memory utilization, disk
space, CPU utilization and/or I/O rates of each SDN controller (2).

8 8. Method according to one of the claims 1 to 7, wherein the step of determining
the best SDN controller (2) comprises using a weighted round robin algorithm.

9 9. Method according to one of the claims 1 to 8, wherein if a plurality of
applications (3) and/or devices (4) intend to connect to the cluster (1) at the same time,
the method comprises
defining, preferably by using a round robin algorithm, a plurality of primary
SDN controllers (2), a different primary SDN controller (2) for each application (3)
and/or device (4),
connecting, by each application (3) and/or device (4), to its primary SDN controller (2),

determining, by each of the plurality of primary SDN controllers (2), one primary SDN controller (2) after the other according to an order, a best SDN controller (2b) for the application (3) and/or device connected to it based on load data of the cluster (1),
instructing, by each of the plurality of primary SDN controllers (2), the application (3) and/or device (4) connected to it to connect to the best SDN controller (2) determined for it,
connecting, by each application (3) and/or device (4), to its best SDN controller (2).

10. Method according to claim 9, wherein determining the best SDN controllers (2) comprises
waiting, by each primary SDN controller (2), for its turn in the order, preferably waiting in a stand-by mode.

11. Method according to claim 9 or 10, further comprising
defining a plurality of secondary SDN controllers (2), a different secondary SDN controller (2) for each application (3) and/or device (4), and
substituting a primary SDN controller (2) exhibiting a failure for its corresponding secondary SDN controller (2b) in all relevant method steps.

12. System comprising a cluster (1) of software defined network, SDN, controllers (2), the system being configured to
define for an application (3) and/or device (4) intending to connect to the cluster (1) a primary SDN controller (2),
connect the application (3) and/or device (4) to the primary SDN controller (2),
determine, by the primary SDN controller (2), a best SDN controller (2) based on load data (6) of the cluster (1),
instruct, by the primary SDN controller (2), the application (3) and/or device (4) to connect to the best SDN controller (2),
connect the application (3) and/or device (4) to the best SDN controller (2).
13. System according to claim 12, further comprising a coordination cluster (5) for storing load data (6) of the cluster (1) using a predetermined path, wherein for determining the best SDN controller (2) the system is configured to lock, by the primary SDN controller (2), said predetermined path of the coordination cluster (5),

read, by the primary SDN controller (2), the load data (6) of the cluster (1) stored in the predetermined path of the coordination cluster (5), calculate, by the primary SDN controller (2), the best SDN controller (2),

update, by the primary SDN controller (2), the load data of the cluster (1) in the coordination cluster (5) according to a load estimation based on the determined best SDN controller (2),

unlock, by the primary SDN controller (2), the predetermined path of the coordination cluster (5),

and reconfigure, by the SDN controller, the application and/or device by the calculated best SDN controller (2).

14. System according to claim 13, being further configured to update, by the best SDN controller (2), the load data (6) of the cluster (1) in the coordination cluster (5), when the application (3) and/or device (4) is connected to the best SDN controller (2).

15. A computer program with a program code for performing a method according to any of claims 1 to 11, when the computer program runs on a computer.
Cluster of SDN Controllers

SDN CONTROLLER (a)  SDN CONTROLLER (b)  SDN CONTROLLER (c)

Locked (waiting)
Lock & read loads + weighted round robin

Devices/Switches

Which SDN controller is best for me?

Coordination Cluster (Zookeeper)

Controller #1: Apps: 2, Devices: 9
Controller #2: Apps: 2, Devices: 8
Controller #3: Apps: 2, Devices: 9

Fig. 4
INTERNATIONAL SEARCH REPORT

A. CLASSIFICATION OF SUBJECT MATTER

INV. H04L29/08

According to International Patent Classification (IPC) or into both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

H04L

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

EPO-Internal, WPI Data, PAJ

C. DOCUMENTS CONSIDERED TO BE RELEVANT

<table>
<thead>
<tr>
<th>Category</th>
<th>Citation of document, with indication, where appropriate, of the relevant passages</th>
<th>Relevant to claim No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>ADVAIT DIXIT ET AL: &quot;Towards an elastic distribut ed SDN control ler&quot;, HOT TOPICS IN SOFTWARE DEFINED NETWORKING, ACM, 2 PENN PLAZA, SUITE 701 NEW YORK NY 10121-0701 USA, 16 August 2013 (2013-08-16), pages 7-12, XP058030691, DOI: 10.1145/2491185_2491193</td>
<td>1-15</td>
</tr>
<tr>
<td>A</td>
<td>US 2012/044532 A1 (TAKAHASI KOUITI ROU [JP]) 23 February 2012 (2012-02-23) the whole document</td>
<td>1-15</td>
</tr>
</tbody>
</table>

Further documents are listed in the continuation of Box C.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

*T* later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"S" document member of the same patent family

Date of the actual completion of the international search

29 January 2014

Date of mailing of the international search report

06/02/2014

Name and mailing address of the ISA

European Patent Office, P.B. 5818 Patentlaan 2
NL - 2280 HV Rijswijk
Tel. (+31-70) 340-2040;
Fax: (+31-70) 340-3016

Pereira, Mafalda
<table>
<thead>
<tr>
<th>Category</th>
<th>Citation of document, with indication, where appropriate, of the relevant passages</th>
<th>Relevant to claim No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>FLORIN BALUS DIMITRI STI LIADIS NUAGE NETWORKS NABI L BITAR VERIZON KENICHI OGAKI KDDI: &quot;Federated SDN-based Controllers for NV03; draft-sb-nvo3-sdn-federation on-01 .txt&quot; ; FEDERATED SDN-BASED CONTROLLERS FOR NV03 ; DRAFT-SB-NVO3-SDN-FEDERATION-01 .TXT, INTERNET ENGINEERING TASK FORCE, IETF; STANDARDWORKINGDRAFT, INTERNET SOCIETY (ISOC) 4, RUE DES FALAISES CH- 1205 GENEVA, SWITZERLAND, 23 October 2012 (2012-10-23) , pages 1-16, XP015085123 , [retrieved on 2012-10-23] the whole document</td>
<td>1-15</td>
</tr>
</tbody>
</table>
## INTERNATIONAL SEARCH REPORT

Information on patent family members

<table>
<thead>
<tr>
<th>Patent document cited in search report</th>
<th>Publication date</th>
<th>Patent family member(s)</th>
<th>Publication date</th>
</tr>
</thead>
<tbody>
<tr>
<td>US 2012044532 Al</td>
<td>23-02-2012</td>
<td>JP 2012043098 A</td>
<td>01-03-2012</td>
</tr>
<tr>
<td>US 2012044532 Al</td>
<td>23-02-2012</td>
<td>US 2012044532 Al</td>
<td>23-02-2012</td>
</tr>
</tbody>
</table>