ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 3, Issue 1, pp: (334-339), Month: January - March 2015, Available at: www.researchpublish.com

Distributed Database Approach for Reconfiguration in Software-Defined Networks

Shantanu Kshire[1], Vishwanath Chikaraddi[2], Jijnasa Patil[3]

[1]P.G. Student, [2]Assistant Professor, [3]Assistant Professor
[1,2,3] Department of Computer Engineering, Veermata Jijabai Technological Institute (VJTI), Mumbai, India

Abstract: **The existing network has many flaws and is incapable of handling Internet. Software-Defined Networking (SDN) is a new way of networking which provides flexibility by decoupling the control plane and the physical plane. SDN brings centralized control over the network but it also increases complexity as network size increases. It will become painstakingly difficult to control the whole network from a single terminal. The solution is to distribute the control of SDN over different sites in order to manage networking infrastructure as a whole. In this paper we have suggested Distributed database system to provide consistent reconfiguration in SDN such that it will reduce the complexity when compared to a centralized approach.**

Keywords: **Distributed Database, Software-Defined Networking, Horizontal Fragmentation.**

I. INTRODUCTION

A. Software-Defined Networking:

Software Defined Networking (SDN) has emerged in order to address the flaws and inflexibility of current networking systems.The current network lacks programmatic control which is provided by SDN. SDN seperates the control plane and management plane of a traditional router from it's data plane. An SDN compatible router consists of only the data plane which results in a unified control and management plane of multiple routers. A controller called as an SDN Controller controls the data plane of multiple SDN routers in order to achieve centralized control over the network. SDN also enables global network view, as result of this developers can programmatically manage the network. However, given its advantages, when considering a huge network consisting hundreds of routers, a centralized approach means there will be large amount of traffic at the controller. There is a need to distribute the control of network from a single SDN Controller located at a single place to multiple locations inorder to simplify the complexity of the traffic. It would mean to assign multiple location administrators, wherein each administrator will be controlling a specific network and making changes to that network would be independent of other networks.

In this paper we suggest a distributed database approach to provide consistent reconfiguration in a Software-Defined Networks. The system consists of a Master database which would be combination of multiple Intermediate databases located at different locations. Along with the Intermediate database each location has it's own Logical database. Along with three types of databases the system also consists of three types of components i.e. Local Component, Global Component and Conversion Component. For simplification at multiple locations we have designed two different databases i.e. Logical and Intermediate DB. The site administrator will specify high-level abstracted configuration of the network to Logical database. The Intermediate DB consits of low-level abstractions. The changes made to Logical DB at each site will trigger local component which will again compare the former low-level abstractions and make changes to Intermediate DB. Multiple Intermediate databases are connected to one Master DB via Global Component. The Master DB initially goes through Horizontal Fragmentation. The task of Horizontal Fragmentation is important as it will introduce fragmentation of database consisting of network information. It will create Intermediate database at different locations. The changes made to Intermediate DB will trigger Global component which will compare the changes made to Intermediate DB with the existing Master Database. Global component also creates an intermediate code will will be used

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 3, Issue 1, pp: (334-339), Month: January - March 2015, Available at: www.researchpublish.com

further by conversion component. New changes will be reflected in the Master DB. The conversion component converts the information stored in intermediate code and generates a configuration file for SDN controller which can be read directly[2]. The conversion component can be flexible i.e. it can create configuration files compatible with different SDN Controller. The main goal of the system is to remain consistent and manage up-to-date information[2] about network configuration from multiple locations. Each component described above can operate independantly and hence can prevent any conflicts caused while information sharing takes place during updations. Further, we also conisder simplified table structures of all the three databases which will be storing network information.

The paper is organized as follows. The information regrading related work is given in Section II. The proposed system is explained in Section III. An example of Horizontal Fragmentation is explained in Section IV. Finally we conclude the paper in Section V.

II. RELATED WORK

The authors in [1] have explained the types of fragmentation and explored Horizontal Fragmentation comprehensively. The Database oriented management for consistent reconfiguration of SDN as decribed in [2] by the authors is a centralized approach for reconfiguration of SDN. In paper [3], the authors have propsed an SDN control system, and claimed that condition of bottleneck takes place on SDN Controllers. The main reason of bottleneck is due to the increase in network size. The same issue also creeps in inconsistency in managing updates since they occur frequently. Authors claimed that due to increase in network size it is hard to maintain consistency of the whole network. Modification of NOX i.e. an SDN Controller is done by authors in [4], inorder to optimize multi-threaded processors and to investigate controller performance in public SDN controllers including their modified NOX.. In [5], a more scalable control system which physically distributes control plane is proposed. Authors claimed that the multiple controller manage different domains of network can create a distributed control system.

III. A DISTRIBUTED DATABASE APPROACH FOR RECONFIGURATION

A. Overview:

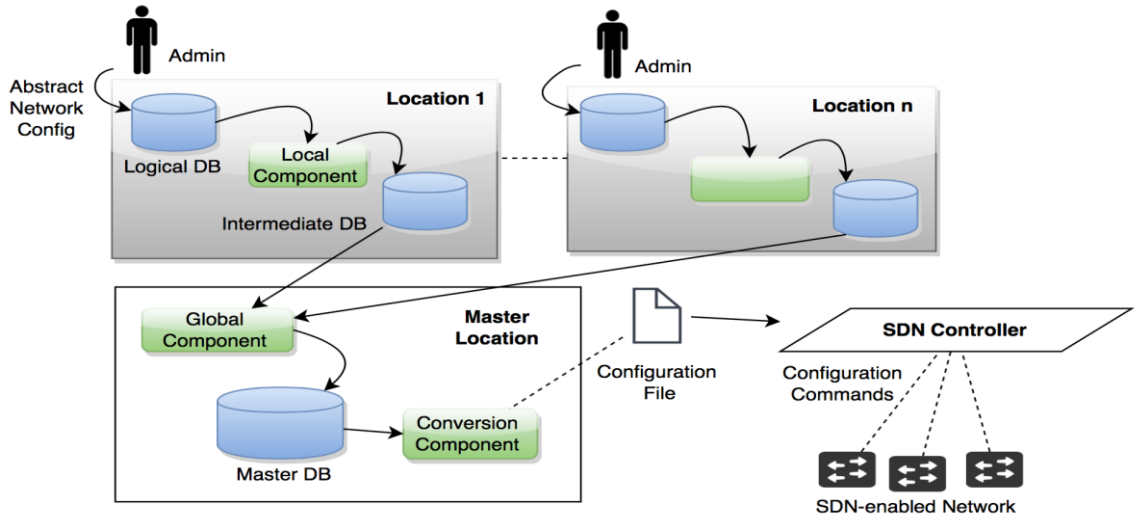

Fig: 1. Overview of proposed system

A simplified overview of distributed database reconfiguration can be given in Fig. 1. Our system consists of three types of databases Logical database,Intermediate database and Master database. Also there are three components namely Local Component, Global Component and Conversion Component. The high-level network abstraction information is provided by a specific site adminitrator to the Logical DB[2]. Based on the information stored in Logical DB the Local component allocates the resources to configure the specific physical network of the assigned location[2]. Then, the low-level network information is stored in the Intermediate DB. This process takes place at each location which is part of the whole network. Global Component creates intermediate code depending on changes made to Intermediate DBs and updates the Master DB. The Conversion Component generates configuration file related to the intermediate code constructed in former

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 3, Issue 1, pp: (334-339), Month: January - March 2015, Available at: www.researchpublish.com

process specific to an SDN Controller[2]. The SDN controller then makes changes to the underlying physical network consisting of switches and routers depending on the configuration file.

B. Databases:

- Logical Database
- Intermediate Database
- Master Database

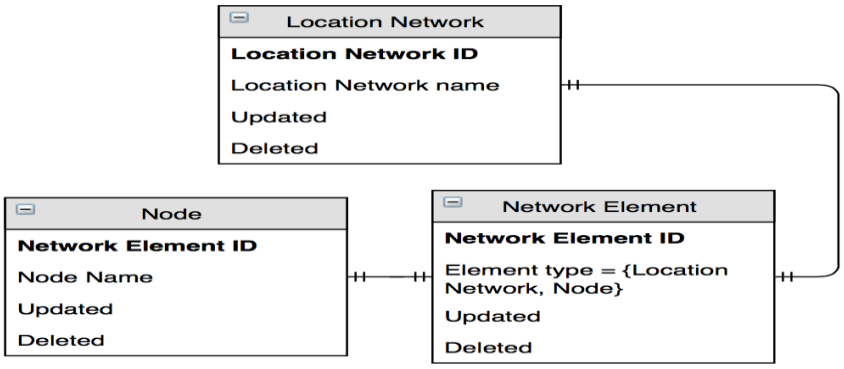

Fig: 2. Logical Database Table structure

Logical Database:

The Logical DB manages the high-level network information specific to network assigned to the location administrator [2]. The Logical DB at each location is independent of each other. Fig. 2. Shows the table structure for Logical DB[2]. The administrator will indicate which node from a list of available nodes for the site will participate in the network, the routing mechanisms adopted as well as the network resources assigned to communication paths. As a result of this the low-level configuration complexity is not seen by the administrator. Any network or a node is a network element and each of them have their seperate Network Element ID.

The location network is managed by the location administrator and is identified by Location Network ID. In Horizontal Fragmentation the Location Network ID(s) are specified to the location administrator. As a result of this any network information related to those location networks will be reconfigured by only the respectively assigned location administrator. From the table structure it is evident that the network can consist of another networks as well. The Updated and Deleted fields are used to check whether the entries have been changed or not[2].

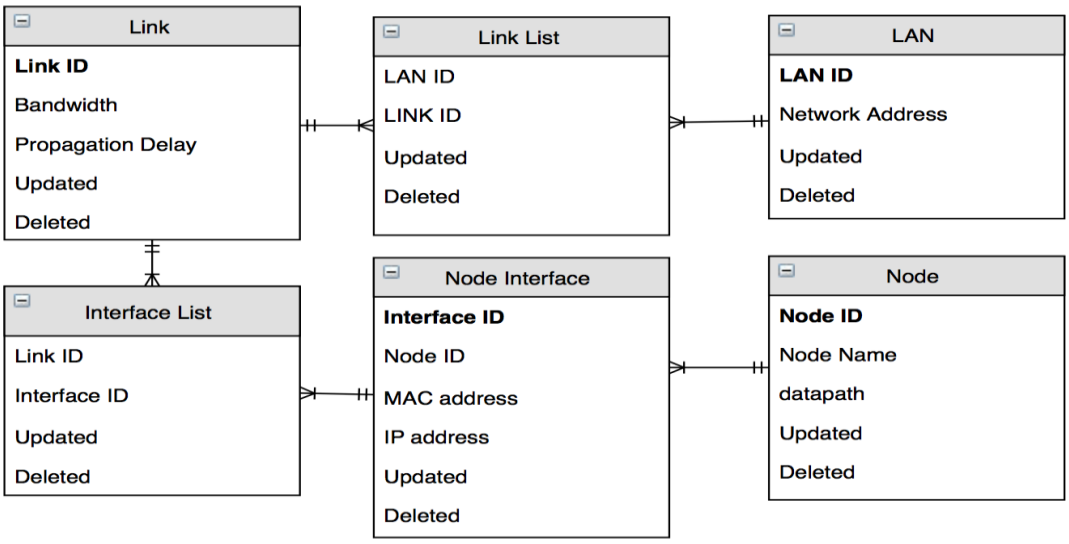

Fig: 3. Intermediate Database Table Structure

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 3, Issue 1, pp: (334-339), Month: January - March 2015, Available at: www.researchpublish.com

Intermediate Database:

The Intermediate DB manages the low-level network information specific to network assigned to the location administrator. The table structure is specified in Fig. 3.[2]. Similar to Logical DB, the Intermediate DBs for each location are independent of each other. By Horizontal Fragmentation of Master DB, Intermediate DB consists of those rows which contain network information assigned to that particular location administrator. The Local Component generates the low-level information for storing in Intermediate DB.It consists of physical information such as node address.

The Intermediate DB consits of total six tables. Link table consits of link information such as delay and bandwidth. LAN table stores the network address and has specific LAN ID alloted for each LAN. The Link list table gives information of links connected to the same LAN. So, a single LAN can have multiple links in it. The node table stores node specific information such as datapath and Node ID[2]. The node interface table manages the physical interfaces of the node[2]. Also a link exists between interfaces for distinct nodes. Interface list table manages all the interface IDs and link ID. So, a node can have many interfaces and many interfaces can have many links. The primary key i.e. the IDs specified in all the tables are Horizontally Fragmented values from the Master DB and thus it is crucial to set them appropriately during creation of the system.

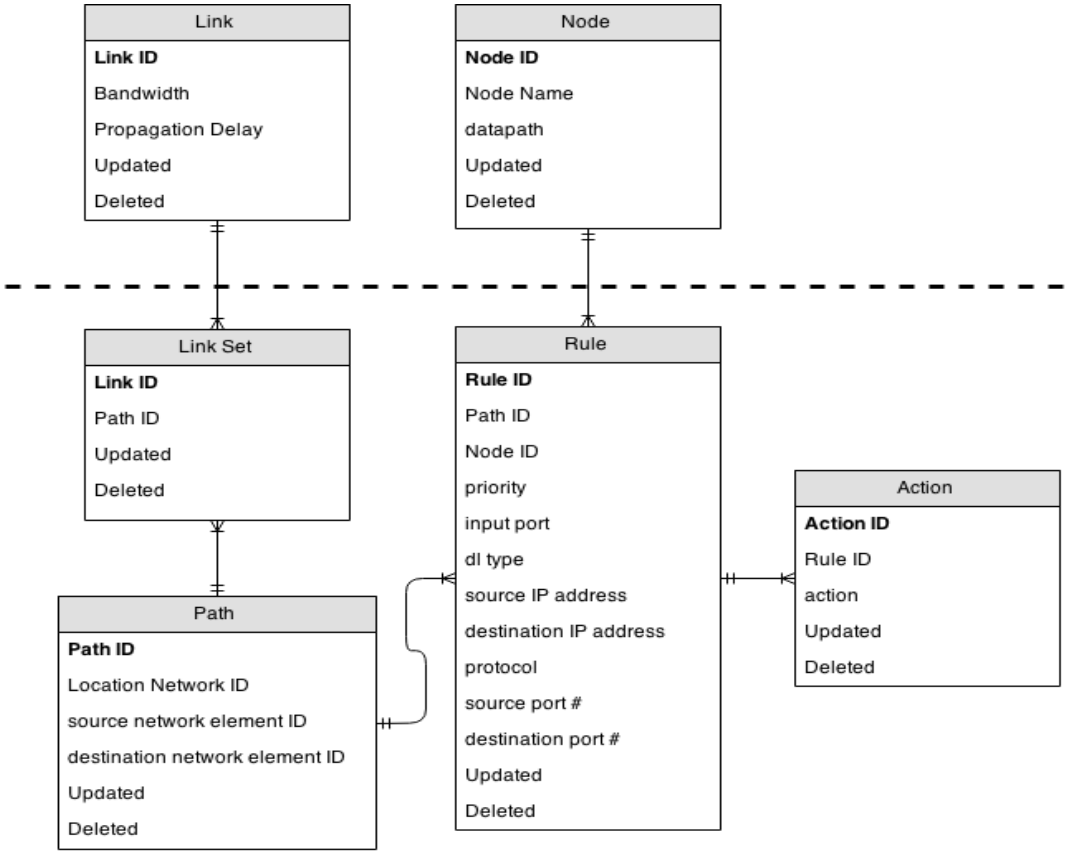

Fig: 4. Master Database Table Structure

Master Database:

The Master DB consisting of the tables in Fig. 4. manages the low-level network information of the whole network[2]. Horizontal Fragmentation takes place at the Master DB. It consits of the same tables as specified in Intermediate DB along with that, Master DB also has Intermediate code table. The Intermediate code tables occupy the part below the dashed line in the figure, while the part above consists of those Intermediate DB tables which have relation with the Intermediate code tables. The main aim is to minimize the load at the Master DB by Horizonatally Fragmenting the network information.

Changes to the Intermediate DB will be fetched by Global component as low-level information to Master DB. In Master DB the changes in Intermediate DB will be attached to the global database consisting of the tables having IDs of the all

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 3, Issue 1, pp: (334-339), Month: January - March 2015, Available at: www.researchpublish.com

the elements. The Global Component also creates an Intermediate code, the intermediate code is fundamental network configuration information essential for configuration of the SDN switches[2]. The intermediate code is managed by the use of four tables. The path table gives the communication path between source and destination participating in the location network which is identified by location ID. The Link set table consists of path ID and those links which are part of the same path. The rule table manages the rules which are to be applied for the switch which is identified by node ID. The Action Table manages switching actions to be applied to the flows. These switching actions are identified at each switch. In short, a path conists of multiple links and multiple rules consist for single path and a single node. Also multiple actions exist for a single rule.

C. Components:

- Local Component
- Global Component
- Conversion Component

Local Component:

The local component is installed locally on each location. The local component fetches the high-level abstracted network information specified by the location administrator. the local component also retrieves the existing low-level network information from the Intermediate DB and compares both, the new abstracted information with the existing low-level network information and makes changes to the Intermediate DB. The local component simplifies the task for global component as it does local processing of high-level information to low-level information reducing the load for global component.

Global Component:

Global component is installed on the main Master Server where the Master DB exists. The Global component listens to all the Intermediate DBs and maps the changes to the Master DB by attaching all the rows in Master DB. The Global component also generates intermediate code essential for further component in the process i.e. Conversion component. The global component uses Master DB in order to manage the intermediate code by the usage of intermediate code tables.

Conversion Component:

The Conversion component converts the information of the intermediate code tables into a suitable configuration file which can be directly read by the SDN controller[2]. The conversion component can be designed flexible such that it can create configuration files for any type of SDN Controller. Conversion component thus makes the distributed database architecture independent of the SDN controller used in the system.

IV. EXAMPLE OF HORIZONTAL FRAGMENTATION

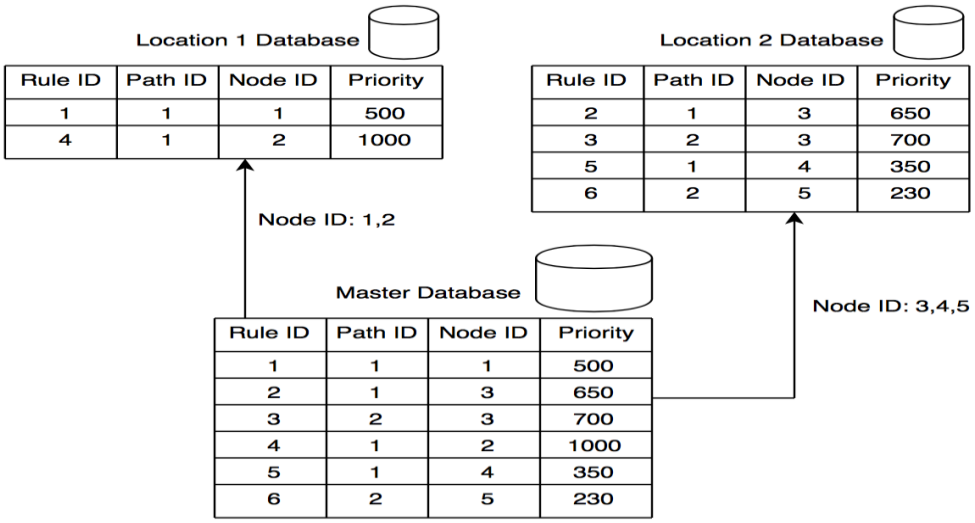

Fig. 5. Example of Horizontal Fragmentation

In the above Fig. 5. we see an example of Horizontal Fragmentation implemented on the Master Database. As different locations will consist of different nodes the Node ID has been decided as the identifier for Horizontal Fragmentation. In the example, Node IDs of 'Location 1' are 1 and 3, while the Node IDs of 'Location 2' are 3,4 and 5. This means that the location administrators of respective locations can only see the Node IDs assigned to them and they can only configure those Node IDs. As we can see that in each location, the Node IDs are independent of each other and any configuration changes made at one location will not disorganize the Node IDs of other location. In this way, there can be 'n' no. of the locations consisting of their respective Node IDs and a global configuration view can be achieved at the Master Database.

III. CONCLUSION

In this paper we have proposed a Distributed database approach for reconfiguration in Software-Defined Networks. We have seen the databases required for different locations and at the master location. The components thus make the system consistent and allow any type of SDN Controller to be compatible with the system. We have also seen an example of Horizontal Fragmentation which can be used to achieve such a system. A decentralized way to reconfigure SDN has been proposed to tackle the problems of centralized systems.

REFERENCES

[1] Ms. P. R. Bhuyar, Dr.A.D.Gawande, Prof. A.B.Deshmukh, "Horizontal Fragmentation Technique in Distributed Database", International Journal of Scientific and Research Publications, Volume 2, Issue 5, May 2012, ISSN 2250-3153.

[2] Yuki Kawai, Yasuhiro Sato, Shingo Ata, Dijiang Huang, Deep Medhi, and Ikuo Oka,"A database oriented management for asynchronous and consistent reconfiguration in Software-Defined Networks", IEEE Xplore Digital Library, 2014.

[3] A. Voellmy and J. Wang, "Scalable software defined network con- trollers," in Proceedings of the 2012 ACM SIGCOMM Computer Communication Review, (Helsinki, Finland), pp. 289–290, August 2012.

[4] A. Tootoonchian, S. Gorbunov, Y. Ganjali, M. Casado, and R. Sher- wood, "On controller performance in software-defined networks," in Proceedings of the 2nd USENIX Conference on Hot Topics in Manage- ment of Internet, Cloud, and Enterprise Networks and Services (Hot- ICE 2012), (San Jose, CA), p. 10, April 2012.

[5] A. Dixit, F. Hao, S. Mukherjee, T. Lakshman, and R. Kompella, "Towards an elastic distributed SDN controller," in Proceedings of the 2nd ACM SIGCOMM Workshop on Hot Topics in Software Defined Network (HotSDN 2013), (Hong Kong, China), pp. 7–12, August 2013.