C-048

OpenFlow based Crossbar Network

Suguru Yasui† Minoru Uehara†

1. Introduction

In recent, renewal computing is important. According to [1], renewal computing is divided two types: rebirth and immortality. Rebirth is corresponding to a like replacement. Immortality is corresponding to sustainability computing and normally off computing. Sustainable computing does not mean only renewable energy based systems but also have environmental adaption and continuous development. As architecture of such a system, we proposed metabolic architecture [2]. In metabolic architecture, it always maintains new condition by exchanging the aggressive. This characteristic is available to large system like a Cloud. We'll implement ARM Linux Cloud as a prototype of metabolic architecture. Elementary node of ARM Linux Cloud (ALC) is Linux installed in ARM Single Board Computer (SBC). However, currently, it is necessary to exchange of elements by manual.

In order to exchange easily, we employ uniform parts。 Parts of ARM SBC employ Raspberry Pi. In order to make Cloud by ARM, it is required to virtualize both hardware and network. Although ARM has no hardware-supported virtualization、but It can be replaced by LXC. In this paper, we virtualize networks using OpenFlow.

In Cloud, clients need to correspond with any servers. In addition, we have to prevent from sniffing packets. In ALC, we organize virtually dedicated path between any client and any server by OpenFlow. Crossbar architecture satisfies such a Requirement. In this paper, it is easily realized by OpenFlow switch of mesh. However, the number of required switches is $O(N)^2$ for the number of servers. It means that the cost of extending network is high. So, we realize VLAN corresponding to crossbar. VLAN can be realized by OpenFlow. The number of switches of VLAN is O(N). However、it have no fault tolerance.

In this paper, we evaluate both crossbar and VLAN methods that are realized using OpenFlow mesh.

2. Design and Implementation of Crossbar

In Cloud, dynamically needs to configuring network between the client and the server. Mesh is the simplest solution that satisfies such requirements. Fig. 1 shows the overview of OpenFlow mesh.

Using a physical architecture of Fig.1, each node for virtualized network is adapted both crossbar and VLAN.

In this mesh, a client is mapped to a horizontal line. A server, which is called a host in Fig. 2, is mapped to a vertical line. Notation in the figure, a client, a switch, and a server as a host are represented as C, S, and H. Also connection of between client C_i and server H_j is C_i-H_j. All of components are ARM SBC (Raspberry Pi).

A crossbar method can be easily realized by mesh network. In this method, in order to connect a client with a server, both a vertical line and a horizontal line need to be occupied. In Fig.1 if communication can be available both a vertical and horizontal line in S_{11}, C_1-H_1 is allowed. Furthermore, if the number of clients or servers is N, the number of required switches is N^2.

† The Graduate School of Info. Sci. and Arts of Toyo Univ.

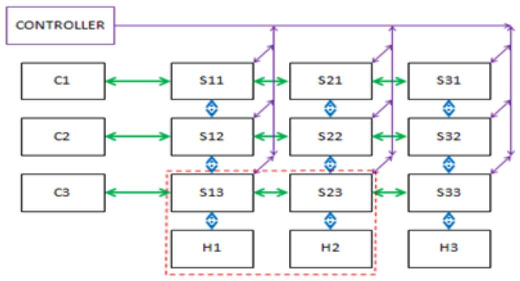

Fig.1 Overview of OpenFlow mesh network

In this paper, a switch is made with Raspberry Pi and 4 USB NICs. The structure of a switch is shown as Fig. 2.

We employ Open vSwitch as OpenFlow switch software. In a switch, a horizontal line is bridged by br1 and a vertical line is bridged by br2. In the initial state, all switches are off. That is, br1 does not connect to br2. For example, in order to connect Ci and Hj, br1 and br2 in Sij will be bridged.

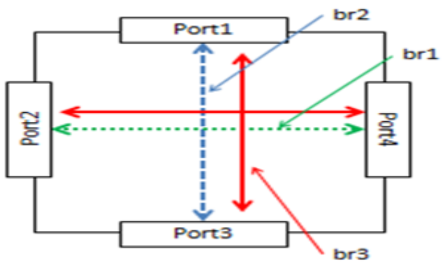

Fig.2 Implement of Open vSwitch

In first condition, vertical and horizontal line is disconnection. For connected C_i and H_j, ports belonged each line are assigned to new bridge in S_{ij}. Thus, crossbar can be realized. In crossbar, the number of required switches is O(N2) where the number of clients or server is N. This is a disadvantage. On the other hand, an advantage is fault tolerance. In crossbar, when a switch is failed, it is easy to assign an alternate path if there is at least a free horizontal line and at least a free vertical line respectively. For example, we assume that Sij is failed. If OpenFlow controller detects the failure, first it finds a free horizontal line Ck and a free vertical line Hl ($l < i$), finally it assigns an alternate path Ci-Sil-Hl-Skl-Ck-Skj-Hj. S1j and SiN are SPF (Single Point of Failure).

3. Design and Implementation of VLAN

An issue on the crossbar is that the number of switches is $O(N^2)$. This is because a line is not shared in crossbar. So, we use VLAN as shown in Fig. 3. Using VLAN, the number of switches is drastically reduced.

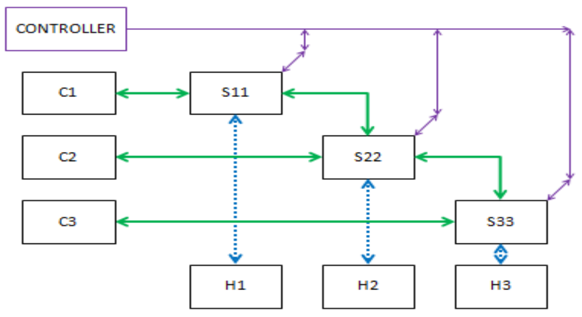

Fig.3 Mesh network using VLAN

The number of required switches is O(N). In VLAN on mesh, only diagonal switches are required. Each diagonal switch is connected to the next switch like linked list. Each line is used by more than one client/server.

In VLAN on mesh, the same switch shown in Fig. 2 is used. In VLAN, br3 is used instead of br1 and br2. If port 1, 3 is labeled to vid 1 and port 2, 4 is labeled to vid 2 respectively, it plays as crossbar.

Using VLAN, the destination of a packet is detected by vid and a line can be shared. For example, in Fig. 3, if C_1-H_2 is requested, all switches' ports From C_1 to H_2 are assigned same vid. Packets distinguish routes by vid. That is, new vid is added to east port and west port in S_{11} and north port and south port in S_{22}. S_{11} flows vid packets horizontally. S_{22} flows vid packets vertically.

An advantage of this method is that the cost is low because the number of required switches is O(N). However, it weakens fault tolerance and performance.

4. Evaluations

Here, we evaluate the performance of switches used in our OpenFlow mesh. In the test bed of OpenFlow mesh, a client, a server and L switches (SWs) are there. Therefore, totally, 3L+2 Raspberry Pies are used. The test bed is illustrated as Fig. 4. Fig. 4 is shown in case of L=2. More than one switches is inserted between a client and a server. The delay will be increased in proportional to the number of inserted switches, i.e. the number of hop.

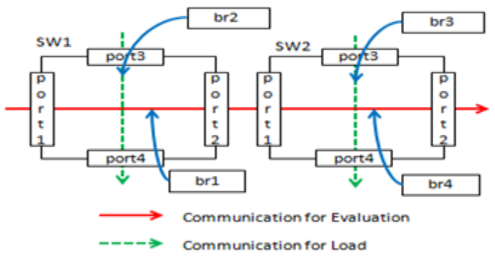

Fig.4 The test bed of OpenFlow mesh

The evaluation is carried out by the following way. The sender (a client) sends 1GB data to the receiver (a server). We measure the time from sending first data to receiving last data (actually sending back the acknowledge). The sender runs "dd if=/dev/zero" and connected to the receiver's command "dd of=/dev/null" via SSH. In Fig. 4, the path from the sender to the receiver is shown as red arrow. During transferring, each switch is given the traffic of transferring 2GB data. This is the system load. The load path is shown as green dashed arrows.

We measure 5 times and calculate the average. Table 1 shows the evaluation results. In table 5, we show 2 kinds of times. The time is measured in case of loaded or not. These values are almost same. Therefore, the system load does not affect the network throughput or a little. In addition, the delay is surely increased in proportional to the number of inserted switches but it is a short. Therefore, the length of path does not affect the network throughput or a little.

Table.1 1GB Data Transfer Times

	Crossbar		VLAN	
L(#SW)	No load	Loaded	No load	Loaded
1	189.1s	188.7s	184.7s	189.7s
2	187.6s	192.3s	187.1s	192.7s
3	183.0s	191.9s	190.5s	193.3s

5. Conclusions

In this paper, we propose OpenFlow mesh based crossbar and VLAN as a network platform of ALC. We construct the prototype of OpenFlow mesh and evaluate its performance. The number of switches required for crossbar and VLAN is O(N2) and O(N) respectively. The difference of performance between both topologies is small. Therefore, VLAN is better than crossbar. From the viewpoint of fault tolerance and renewability, we cannot conclude that it is always true. However, the difference between the costs of both topologies is large. Therefore, more than one alternative path should be added to VLAN in order to increase fault tolerance. Finally, the maximum number of vids is limited. So, VLAN is not final solution.

In future, we will employ OpenFlow switch, which can realizes to share the path without VLAN. In OpenFlow switch, IP based routing is possible. IP addresses of clients and servers are given by cloud services. In this way, using OpenFlow, cross-layered routing is possible. Furthermore, we will discuss on fault tolerant routing. And, we will integrate switches and servers. In our architecture, both elements are made with Raspberry Pi. Therefore, there is no reason why switches differ from servers. If all switch become a server, fault tolerance of such a system is very high. Finally, we will build automatic renewable cloud using ALC.

References

[1] Minoru Uehara: "Research Trends on Renewable Computing System -Reports from WReCS-2013 Workshop-", IEICE TR RIS No.8, Vol.6, pp.1-6, (2013.10.19) (in Japanese)
[2] Minoru Uehara: "Metabolic Computing", In Proc. of the 5th International Workshop on Advanced Distributed and Parallel Network Applications(ADPNA2011) in conjunction with the 14th International Conference on Network-Based Information Systems(NBiS2011), pp.370-375, (2011.9.7-9,Tirana,Albania)
[3] Suguru Yasui, Minoru Uehara: "OpenFlow Mesh based Virtual Crossbar Network", CISIS2016, (TBA).
[4] OpenFlow Networking Foundation: "OpenFlow Switch Specification Version 1.5.1"