SRP: A Routing Protocol for Data Center Networks

Pengcheng Zeng*, Yao Shen*, Zijun Qiu†, Zhun Qiu†, Minyi Guo*
*Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, 200240 China
†Tencent Technology (Shenzhen) Company Limited, Shenzhen, 518057 China
Email: pencheil@sjtu.edu.cn, yshen@cs.sjtu.edu.cn, treeqiu@tencent.com, zhunqiu@gmail.com, guo-my@cs.sjtu.edu.cn

***Abstract*—Open Shortest Path First (OSPF) is a broadly used routing protocol in today's data center networks. Though it can adaptively find alternative paths when link failure is detected, route recomputation in OSPF is CPU-intensive, and usually needs quite a long time to achieve routing convergence. Besides, the configuration in OSPF is rather complex.**

In this paper, we propose Sequoia Routing Protocol (SRP), a routing protocol with a dedicated shortest path calculation algorithm, designed for data center networks. By leveraging the observation that each data center has a fixed Clos topology, SRP largely simplifies routing configuration, and achieves light-weight computation. We show that SRP outperforms OSPF with a much faster convergence time by simulation in ns-3. We also implement SRP in real-world switches, showing that SRP is as efficient as OSPF even for a small-scale network where route recomputation of OSPF does not take a long time.

***Index Terms*—Data Center Network, SRP, OSPF.**

I. Introduction

Open Shortest Path First (OSPF) is one of the mainstream routing protocols for data center networks. However, the time complexity of Dijkstra algorithm used by OSPF is $O(V^2)$ ($O(V*lgV+E)$ if using Fibonacci heap) when V is the number of vertices in the graph. Due to large amounts of shortest path first (SPF) calculations, CPU utilization will be rather high. Moreover, it has a long convergence time [1] when link state changes (e.g., link failure); such ineffectiveness can cause a large number of packet losses [2]. OSPF also suffers from the link state scaling problem. As a link-state routing protocol, OSPF maintains a link state database (LSDB) to store the topological information about the network, which is used for route calculation. Consequently, as network grows, more memory will be consumed, LSDB may even become oversized. A final and equally important point is that configuration in OSPF is rather complicated, time-consuming and error-prone. To sum up, OSPF is not well-suited in today's large-scale data center networks, and should be replaced to make performance better and configuration more straightforward.

In this paper, we introduce Sequoia Routing Protocol (SRP), a routing protocol specially designed for data center networks. The design of such a routing protocol is made challenging by dynamic changes in the network, which includes both the frequent failures in switch interfaces and links (intermittent or permanent, caused by software or hardware faults), as well as network congestion on some switches with dynamic network flows. Besides, network maintenance and provisioning events may also result in network changes. In the design and implementation of SRP, we leverage the observation that the topology of data center networks is usually a simple and fixed Clos [3][4][5]. Routing paths in such a data center network are countable. To keep track of link states in each switch, instead of LSDB in OSPF, we use Grid, a routing database for SPF calculation. Grid is a table with a fixed size, in which the value of each cell is automatically preset by SRP Controller. Like in OSPF, a link state advertisement (LSA) is broadcast when changes in table are detected. SPF calculation can be finished within $O(mn)$ time, where m is the number of rows in the table (the total number of subnets), and n is the number of columns (the total number of peers[1]) in Grid. Compared to OSPF, SRP has the following advantages. By avoiding a large amount of invalid route computation, SRP achieves light-weight computation. SRP is also operation-oriented without complex configurations, which has the benefit of fast transplant. Moreover, routes are under control thanks to clear division of labor with the assistant of SRP Controller.

The key contributions of our work are:

- We design and implement SRP, a routing protocol specifically for data center networks, simplifying routing configuration and achieving light-weight computation with a dedicated SPF algorithm.
- We demonstrate that SRP outperforms OSPF in ns-3.
- We implement SRP in physical switches, showing that SRP is as efficient as OSPF even for small-scale networks where route recomputation of OSPF can be ignored.

The rest of this paper is organized as follows. Section II discusses about related researches with our work. Section III gives an overview and an analysis about the data center network topology. Section IV describes a detailed design of SRP. We evaluate SRP in Section V with a simulation in ns-3 and a real-world implementation. Section VI concludes our work.

II. Related Work

This section surveys previous work on data center networks.

Greenberg et al. realizes Monsoon [6], using programmable commodity layer-2 switches and servers to set up a simple mesh-like architecture. Dcell [7] and BCube [8] both propose specific structures for data centers, working on different network topologies with their own forwarding strategies. Our work, SRP, is built on a Clos-like topology, which have been widely applied in today's data center networks.

[1]We will use the term *peer* throughout this paper to refer the neighboring switches of some switch

Much work has focused on lower layers for data centers. SEATTLE [9] uses the global switch-level view to form one-hop DHT for address resolution. Mohammad et al. [10] use a two-level routing table and a flow classification scheme for routing. They use a central route control to statically generate all routing tables. They update routes just by disabling or enabling some entries in the routing tables which may be too large to maintain and has poor scalability. VL2 [4] uses flat addressing, Valiant Load Balancing and end-system based address resolution to meet three objectives: uniform high capacity, performance isolation and layer-2 semantics in the Clos network topology. PortLand [5] is a layer-2 data center network fabric proposed by Radhika et al. PAST [11] is a scalable data center architecture on Ethernet with a Per-Address Spanning Tree routing algorithm.

There are a lot of works done on routing protocols redesign. Using precomputed backup routes, like MPLS Fast-Reroute (see in RFC4090) and IP restoration [12], is not a good idea for large-scale data center networks. FCP [13] is a new routing paradigm proposed to discover a working path without completely up-to-date topology state. FCP is applied in WAN. Some researches discussed about adjusting timing parameters to achieve a better performance. However, this is not an substantial improvement, which can also apply to SRP. As far as we know, no related works on routing protocols have been undertaken to replace OSPF in data center networks.

Hedera [14] is a flow scheduling system which can significantly outperform the state-of-the-art hash-based ECMP load-balancing. A better flow scheduling mechanism with congestion control, is what SRP deficient in. With the development of SDN and OpenFlow [15], like RouteFlow [16], we can also implement SRP in a centralized way to achieve performance improvement and full route control.

III. Overview

In this section, we give a detailed analysis on a typical data center network topology that SRP will be applied.

A. Data Center Network Topology

We use a two-level tree of switches (see Fig. 1) as our data center network architecture, a typical Clos structure [3]. As shown in Fig. 1, the network is a variant of fat-tree [17]:

- 512 Top of Rack (ToR) switches[2]. Currently, a standard ToR switch contains 1 to 4 10GigE uplinks and 48 1GigE downlinks [5].
- 4 core (Core) switches for interconnection of ToR switches. The number of Core switches will range from 2 to 4 based on the uplink's bandwidth requirement of ToR switches. Some existing Core switches have more than 288 40GigE ports, which can be split to 1152 10GigE ports, fulfilling the need of ToR switches.
- 2 border (Border) switches, managing traffic exchange between data centers. Border switches are almost the same with ToR switches in the topology.

[2]We will use the term *switch* throughout this paper to refer to devices that can be used for both Layer-2 switching and Layer-3 routing.

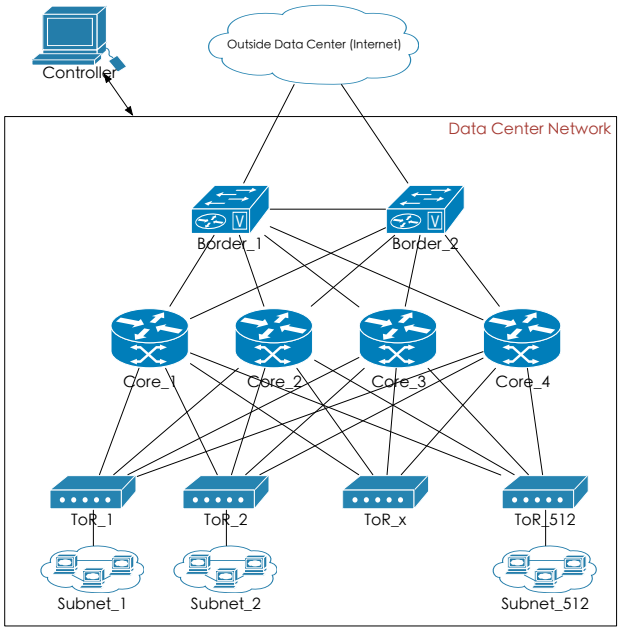

Fig. 1: Data Center Network Topology

Which should be mentioned here is that our design is not limited to this simple two-tiered topology, it also applies to many other more complex fat-tree like topologies (e.g., the three-tiered topology with an aggregation layer), even to some non-fat-tree like topologies. With small changes of Grid (extend some additional types of Grid), SRP can work. Besides that a two-tiered topology is commonly used in current data centers, it also makes our design more clear to comprehend. Such a topology gives more prominence to the benefit of SRP.

Typically, this two-tiered design can support at most 20K servers, which is large enough for most data centers. The flows between any two ToR switches or a ToR switch and a Border switch can represent all network traffic related to the data center. Therefore, the shortest path between any pairs will definitely pass one of the Core switches. When routing with the extension of Equal Cost Multipath (ECMP), the amount of paths is equal to the amount of Core switches. Consider such a fat-tree as a combination of multiple trees, each rooted with a Core switch, we can call it Multi-Root-Tree in which it will be rather simple to compute and maintain shortest paths.

The data center network architecture in Fig. 1 can be split into four independent trees, each tree with one Core switch. It is a easy way to calculate the shortest paths in each tree using static addressing. To add together, or stack, all the shortest paths of the 4 trees, we can get those of the whole topology. In addition, because ToR switches are shared by four trees, it is necessary to Vertical Split (forwarding a flow from one Core switch to another is not allowed when there are other shorter paths between the two corresponding ToR switches), which can be implemented by fixing optional next hop of each IP subnet in Core switches.

TABLE I: SRP Grid: a preset routing database, composed by Core Grid, ToR Grid and Border Grid.

I.1 Core_1 Grid

Subnets	ToR1	ToR2	...	ToR128	Border1	Border2	Description
10.1.1.0/26	up	invalid	...	invalid	invalid	invalid	T1_Production
10.1.1.64/26	invalid	up	...	invalid	invalid	invalid	T2_Production
.....	...	...	...	...	...	...	
10.1.32.192/26	invalid	invalid	...	up	invalid	invalid	T128_Production
0.0.0.0/0	invalid	invalid	...	invalid	up	up	B_exit

I.2 ToR_1 Grid

Subnets	Core1	Core2	Core3	Core4	Description
10.1.1.64/26	up	up	up	up	T2_Production
10.1.1.128/26	up	up	up	up	T3_Production
.....	...	...	...	...	
10.1.132.192/26	up	up	up	up	T128_Production
0.0.0.0/0	up	up	up	up	B_exit

I.3 Border_1 Grid

Subnets	Core1	Core2	Core3	Core4	Border2	Description
10.1.1.0/26	up	up	up	up	backup	T1_Production
10.1.1.64/26	up	up	up	up	backup	T2_Production
.....	...	...	...	...		
10.1.32.192/26	up	up	up	up	backup	T128_Production
10.1.0.1/32	up	invalid	invalid	invalid	backup	C1_Loopback
10.1.0.2/32	invalid	up	invalid	invalid	backup	C2_Loopback
10.1.0.3/32	invalid	invalid	up	invalid	backup	C3_Loopback
10.1.0.4/32	invalid	invalid	invalid	up	backup	C4_Loopback

B. Analysis

In addition to the characteristic that it is known and simple, a data center network topology is also constant with a relatively fixed size. ToR switches with a fixed IP subnet are ordered by an index starting from 1, and normally they remain unchanged in the data center. Thus, these subnets can be calculated by a formula following the data center construction plan. Subnets are aggregated into one route for Border switches controlling traffic exchange between the inside and the outside of the data center. Border switches are responsible for publishing default routes to the inside of the data center.

We leverage the observation that the routes inside the data center is relatively constant, and what changes frequently is the state of each subnet:

- When ToR is off, the corresponding subnet is Down;
- When ToR is on, the corresponding subnet is Up;
- When a link between ToR and Core is disconnected, the corresponding subnet is Down in the tree whose root is the Core switch.

Accordingly, compared with OSPF, SRP simplifies routing and forwarding in data center networks by: 1) Presetting some routing information (Grid, which will be mentioned later) statically for SPF in each tree; 2) Denying routes that are not preset; 3) Maintaining peer relationships among switches; 4) Monitoring default subnets in routing tables, and communicating the states with peers; 5) Modifying default static routing states when a update message is received or peer state changes.

IV. Design And Implementation

A. Routing Database: SRP Grid

The key insight behind SRP is SRP Grid, a preset routing database. Each switch has an independent SRP Grid. As shown in Table I, Grid is a table, whose columns represent SRP peers and rows represent destination IP subnets. Each cell in Grid represents the availability of the corresponding subnet via the corresponding peer, "invalid", "backup", "up" and "down". "invalid" indicates that the subnet can not be reached, and will not change all the time. "up" indicates that it's now available, and "down" is the opposite: it's temporarily unavailable. They are SRP's default values.

There are three general types of Grid: Core Grid, ToR Grid and Border Grid in data center networks. Core Grid is brief: except for the default route (0.0.0.0/0), there is only one cell in a row whose value is "up" or "down". This is a key feature of packet forwarding in Fat-Tree topology, avoiding computing invalid routes. ToR Grid is similar with Border Grid. Each cell in a row represents a forwarding path in a different tree. In Border Grid, there are cells whose values are "backup". The specificity just lies in Border Grid, for backup path when a packet is forwarded in from the outside of the data center. Only when there is no cell whose value is "up" for a specified subnet (there is no optimal next hop for that subnet) will this cell come into operation.

Switches load their Grid configurations sent from SRP Controller at startup time (at this time, the cost of making and dispatching Grids is negligible), and alter values of the cells when network changes. One situation that causes the change is when SRP detects disconnection of its peer, all the cells whose value is "up" in the corresponding column will be changed to "down". When receiving a Link State Advertisement (LSA), SRP will process as follows:

1) Check if its local Grid contains the subnet that is in the LSA; if not, discard that message and stop;
2) Check if the value of the corresponding cell is "invalid"; if so, discard it and stop;
3) Check if the value is consistent with the state in the message; if so, keep the value unchanged and stop;
4) If the value is "up", change it to "down"; or to "up".

The change of the value of a cell from "down" to "up" will generate a static route; and conversely, it will revoke this one.

Algorithm 1 Route Calculation in SRP

```
1: for each subnet s in Grid do
2:   for each peer p in s do
3:     Cvalue =value of the corresponding cell
4:     if Cvalue is up then
5:       add p to the routing table
6:     end if
7:   end for
8: end for
```

Algorithm 1 shows the working process of route calculation in SRP. As with OSPF, we also extend SRP with ECMP to make full use of mutliple paths in the topology. SRP works for a large-scale data center, with a similar and fixed layout of IP address and network topology. Subnets and peers should be well planned in advance, not with the pace of construction. If the plan changes, adding or deleting a subnet or a peer, Grids in all switches should be updated and reloaded.

B. SRP Message

Change of SRP peer relationship is the only way to alter the value of a cell. There are 3 types of peer messages: Open, Keepalive and Update. Update message can be further divided into two types: one with a list of invalid subnets, another with a list of valid subnets. By default, SRP maintains its peer infomation via Keepalive messages; SRP can also use Bidirectional Forwarding Detection (BFD) to speedup failure detection. Open message requires IP addresses of peers and its corresponding source interface, together with switch type (Core, ToR or Border) and switch identifier (the index of switches, e.g., 1, 2, 111 and so on) for authenticating. As an IGP, multi-hop is not recommended.

SRP will check state change of each subnet in Grid. When change occurs, Update message will be generated and processed as followed:

1) When the subnet is unavailable – there is no cell whose value is "up" (except that cell with value is "backup") in the corresponding row, Update message will be broadcast to notify that the subnet loses its effectiveness;
2) When there are more than one cells whose value is "up" all the time, no Update message will be generated since the subnet is always reachable;
3) When the number of cells whose value is "up" changes from 0 to 1 or more, then Update message will be broadcast to notify that the subnet takes effect.

C. Aggregation Routing

In conventional data center networks, routes inside the data center are transmitted by the more specific routes, while routes outside the data center will be aggregated to minimize the number of routing tables required among different data centers [18]. As in Fig. 2, the more specific routes will be published between Cores and ToRs, and routes between Cores and Borders will be aggregated.

However, there is one problem existing for route aggregation. Assume there is a flow from Subnet2.1 to Subnet1.1 going through ToR2.1, Core2.1, Border1, Core1.1 and ToR1.1 orderly. When link between Core1.1 and ToR 1.1 goes down, Border1 will not converge since it only have the information of aggregation routes, and changes inside the data center will not be known. No other new path will be established. OSPF will then route packets from Core1.1 to ToR1.2 (or ToR1.3), Core1.2 and then to ToR1.1, leading to large amounts of routing loops. While in SRP, with the coordination of SRP Controller, Core1.2 will send routes of Subnet1.1 to Borders separately. And Borders will modify its routing tables and forward flows for Subnet1.1 to Core1.2 using the longest prefix match, thus achieving a better routing path. SRP is the only routing protocols known which have now implemented this detailed scheduling mechanism.

V. Evaluation

We evaluate SRP using both simulations and a real-world implementation.

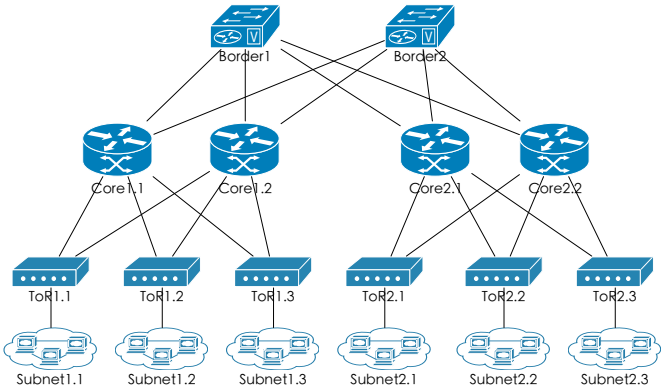

Fig. 2: Route aggregation with two data centers

A. ns-3 Simulation

We implemented both SRP and OSPF in ns-3. ns-3 is a discrete-event network simulator for Internet systems, built in C++ and Python. Any computation in ns-3, including packet processing and SRP/OSPF handling, will not count. They will be finished at once. This is inadequate for our evaluation. Hence we included real-world processing delays to extend ns-3. We used Arista 7050S-64 switch, a mainstream platform in the world's largest data centers. The Arista 7050S-64 switch runs Arista EOS, a modular switch operating system supporting Linux. Thus, we can run C/C++ programs on Arista EOS.

OSPF contains delays for transfer packets, detecting link/switch failure, processing LSAs, performing SPF calculations, updating FIB and flooding LSA [19]. In our ns-3 evaluation, SRP has a similar working mechanism with OSPF except route calculation.

We chose 1000ns as packet latency in ns-3 [20]. We could achieve fast failure detection using BFD within sub-20ms. The time for LSA processing is fast ($\leq$ 12ms), and in our experiment, there is just one LSA to flood in which the time is negligible [21]. The FIB update is not a key factor which distinguishes SRP from OSPF, and that is the hardware mechanism for routing and forwarding speedup (see Section III-B). We did not implement FIB mechanism in ns-3, so the duration could be regarded as 0ms. *spf-initial-wait* (initial LSP generation delay) and *spf-hold* (minimum hold time between two consecutive SPF calculations) are parameters in OSPF which could be set manually to different values. To ease the comparison between simulations, we just set these parameters to be 0ms.

We tested route calculation delay of SRP and OSPF on Arista 7050S-64 switch. The route calculation includes the process of SPF computation and computation from Grid/LSDB to routing tables. We ran the testing programs 50 times on the switch with the number of ToRs ranging from 4 to 1500 and the number of Cores ranging from 2 to 4. Figures in Fig. 3 show the delay, the left one is average delay time in Core switches, and the right one is that in ToR switches. These two figures accord with the computation complexity we got

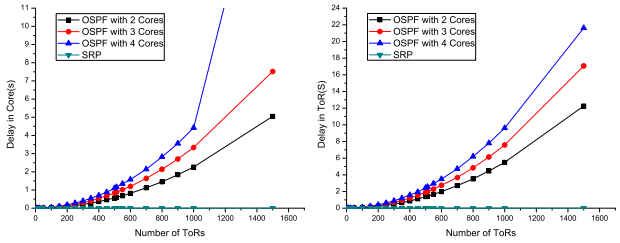

Fig. 3: Route calculation delay in SRP and OSPF

TABLE II: Simulation parameters

parameter	time
packet-latency	1 ms
failure-dection-delay	20 ms
lsa-process-delay	10 ms
flooding-delay	0 ms
FIB-update-delay	0 ms
spf-initial-wait	0 ms
spf-hold	0 ms
route-calculation-delay	see Fig. 3

before. In a typical data center network topology, the number of Core switches ranges from 2 to 4. When the number of ToR switches is below 10 thousand, there is little delay of SRP (almost 0 ms) for both Core and ToR switches; however, the delay of OSPF increases both with the number of ToR switches and Core switches in super-linear rate. For example, when the number of Cores is 4 and the number of ToRs is 512, the average route calculation delay of SRP is 2.56 s. We can prove the idea mentioned in Section I that the routing recomputation in SRP is light-weight, which wins out over OSPF greatly. We set route calculation delay by reference to Fig. 3.

The algorithms for SPF computation are the main difference between SRP and OSPF. We tried our best to minimize the influence of other parameters. Table II reports the parameters in the ns-3 simulations of SRP and OSPF. The parameter route-calculation-delay changes with the number of Core and ToR switches.

We extended ns-3 with a SRP module and a OSPF module with parameters in Table II. And we extended Ipv4RoutingProtocol class in ns-3, implementing RouteInput and RouteOutput member functions, and modified Ipv4L3Protocol and Ipv4ListRouting classes for delay simulation and information analysis.

We built up a topology as Fig. 1 running SRP and OSPF separately, and created a UDP network flow. The flow originated from a server A in Subnet_1, and its destination is another server B in Subnet_2. The convergence time will not be affected by packet size and sending rate according to the equation below. Compared with TCP, UDP is a connectionless transportation protocol with no heavy-weight resending mechanism. It is a better choice for packet loss measurement to use

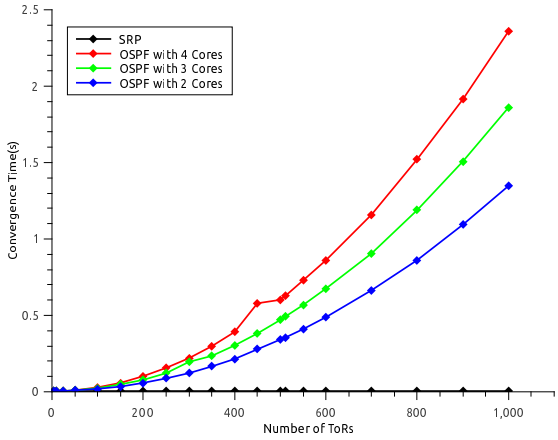

Fig. 4: Convergence time with SRP and OSPF

TABLE III: OSPF and SRP Result in real-time networks

	ToR_1 → ToR_2	ToR_2 → ToR_1
OSPF Link Down	0.55s	0.51s
OSPF Link Recovery	0.00s	0.08s
SRP Link Down	0.56s	0.50s
SRP Link Recovery	0.00s	0.00s

UDP.

Convergence time is the time taken by the switches to go back to a steady state after a state change in the network [1]. We compare SRP against OSPF in terms of convergence time for our benchmark, which can be computed by:

$$Convergence\ Time = \frac{Packets\ Lost \times Packet\ Size}{Sending\ Rate}$$

Because of the feature of ns-3 (discrete-event), with fix parameters, the results getting from ns-3 are almost the same. Even though, we ran the experiments 50 times and got the average convergence time. Fig. 4 shows the result of our experiments in ns3. When the number of ToRs is below 5000 (which is almost the largest size of existing data centers), the convergence time of SRP is rather low (4.9ms), while that of OSPF is increasing with the number of ToRs. Take 4 Cores and 512 ToRs for example, which is a typical data center's scale. The convergence time of OSPF is 629.9ms, 128 times slower than SRP.

B. Real-world Implementation

We implemented SRP in read-world switches and set up a small-scale network for testing.

Limited by the number of switches, we set up a topology with two Cores and two ToRs for testing. There are stable flows between ToR_1 and ToR_2. What we did is to shut down the link between ToR_2 and Core_1, and then recover this link in a few minutes. We recorded the packets dropped after this two actions. We also used the formula above to calculate the convergence time. The result is shown in Table III.

When link down, the flow from ToR_1 to ToR_2 will recover after: (1) Core_1 detects that the link is down, (2) Core_1 sends Update message to ToR_1, (3) ToR_1 refreshes Grid, (4) ToR_2 modifies routing table, (5) ToR_1 updates

FIB; while what the flow from ToR_2 to ToR_1 need to do is just to update FIB. Actually, there will be some delay in real-world switches between RIB and FIB update. FIB update will not immediately take effect after RIB (routing table) update. Besides, since the current implementation of SRP on real-world switches has not been optimized to the best, the convergence time when link down is almost the same as OSPF. However, we did prove that in real-world switches, SRP is suitable for such a Clos-like network. As the number of switches increases, the advantage of SRP over OSPF will be highlighted.

The experiment on real-world switches shows that SRP is comparable with OSPF even for a small-scale network where route recomputation of OSPF does not take a long time.

VI. Conclusion

We proposed Sequoia Routing Protocol (SRP), a routing protocol designed for data center networks. Unlike LSDB in OSPF, we defined Grid, a routing database for SPF computation in SRP. By taking advantage of data center's fixed and simple Clos topology, SRP simplifies routing configuration and achieves light-weight computation with a dedicated shortest path calculation algorithm. We also empirically demonstrated the advantages of SRP over OSPF both in ns-3 and real-world switches. In a typical data center network (4 Cores and 512 ToRs) that we set up in ns-3, we achieved a significant performance improvement: the convergence time is 128 times faster than that of OSPF. In a real-world environment, even for a small-scale network where route computation of OSPF can be ignored, we showed that SPR still worked as efficiently as OSPF.

In the future, we will extend SRP with a congestion control mechanism. Another interesting direction we are considering now is to embed SRP into OpenFlow, which can decouple route computation from the hardware switches and enable remote routing services in a centralized way.

Acknowledgements

We would like to thank Tencent Technology (Shenzhen) Company Limited (especially Miaoxin Cai, Minghua Yi, Quan He, Yi Ding), and Huakang Li, to give us help in this paper. This work was partially supported by Shanghai Excellent Academic Leaders Plan (No. 11XD1402900), Program for Changjiang Scholars and Innovative Research Team in University (IRT1158, PCSIRT) China, NSFC (Grant No. 61272099, 61261160502) and Scientific Innovation Act of STCSM(No.13511504200). Yao Shen is the corresponding author.

References

[1] Anindya Basu and Jon G. Riecke. Stability issues in OSPF routing. In *Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*.

[2] Phillipa Gill, Navendu Jain, and Nachiappan Nagappan. Understanding network failures in data centers: measurement, analysis, and implications. In *Proceedings of the ACM SIGCOMM 2011 conference*, pages 350–361, 2011.

[3] Charles Clos. A Study of Non-Blocking Switching Networks. In *Bell System Technical Journal*, volume 32, pages 406–424, 1953.

[4] Albert Greenberg, James R. Hamilton, and Navendu Jain. VL2: A Scalable and Flexible Data Center Network. In *Proceedings of the ACM SIGCOMM 2009 conference on Data communication*, pages 51–62, 2009.

[5] Radhika Niranjan Mysore, Andreas Pamboris, Nathan Farrington, Nelson Huang, Pardis Miri, Sivasankar Radhakrishnan, Vikram Subramanya, and Amin Vahdat. PortLand: a scalable fault-tolerant layer 2 data center network fabric. In *Proceedings of the ACM SIGCOMM 2009 conference on Data communication*, pages 39–50, 2009.

[6] Albert Greenberg, Parantap Lahiri, David A. Maltz, Parveen Patel, and Sudipta Sengupta. Towards a Next Generation Data Center Architecture: Scalability and Commoditization. In *Proceedings of the ACM workshop on Programmable routers for extensible services of tomorrow*, pages 57–62, 2008.

[7] Chuanxiong Guo, Haitao Wu, Kun Tan, Lei Shi, Yongguang Zhang, and Songwu Lu. Dcell: a scalable and fault-tolerant network structure for data centers. In *Proceedings of the ACM SIGCOMM 2008 conference on Data communication*, pages 75–86, 2008.

[8] Chuanxiong Guo, Guohan Lu, Dan Li, Haitao Wu, Xuan Zhang, Yunfeng Shi, Chen Tian, Yongguang Zhang, and Songwu Lu. BCube: a high performance, server-centric network architecture for modular data centers. In *Proceedings of the ACM SIGCOMM 2009 conference on Data communication*, pages 63–74, 2009.

[9] Changhoon Kim, Matthew Caesar, and Jennifer Rexford. Floodless in seattle: a scalable ethernet architecture for large enterprises. In *Proceedings of the ACM SIGCOMM 2008 conference on Data communication*, pages 3–14, 2008.

[10] Mohammad Al-Fares, Alexander Loukissas, and Amin Vahdat. A scalable, commodity data center network architecture. In *Proceedings of the ACM SIGCOMM 2008 conference on Data communication*, pages 63–74, 2008.

[11] Brent Stephens, Alan Cox, Wes Felter, Colin Dixon, and John Carter. Past: Scalable ethernet for data centers. In *Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies*, CoNEXT '12, pages 49–60, New York, NY, USA, 2012. ACM.

[12] Iannaccone Gianluca, Chuah Chen-Nee, Bhattacharyya Supratik, and Diot C. Feasibility of IP restoration in a tier 1 backbone. In *Network, IEEE*, pages 13–19, 2004.

[13] Karthik Lakshminarayanan, Matthew Caesar, Murali Rangan, Tom Anderson, Scott Shenker, and Ion Stoica. Achieving convergence-free routing using failure-carrying packets. In *Proceedings of the 2007 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 241–252, 2007.

[14] Mohammad Al-Fares, Sivasankar Radhakrishnan, Barath Raghavan, Nelson Huang, and Amin Vahdat. Hedera: dynamic flow scheduling for data center networks. In *Proceedings of the 7th USENIX conference on Networked systems design and implementation*, pages 19–19, 2010.

[15] Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner. OpenFlow: Enabling Innovation in Campus Networks. In *Proceedings of the ACM SIGCOMM 2009 conference on Data communication*, pages 39–50, 2009.

[16] Marcelo R. Nascimento, Christian E. Rothenberg, Marcos R. Salvador, Carlos N. A. Corrła, Sidney C. de Lucena, and Maurcio F. Magalhes. Virtual routers as a service: the RouteFlow approach leveraging software-defined networks. In *Proceedings of the 6th International Conference on Future Internet Technologies*, pages 34–37, 2011.

[17] Charles E. Leiserson. Fat-trees: universal networks for hardware-efficient supercomputing. *IEEE Transactions on Computers*, 34(S2):892–901, October 1985.

[18] E. Chen and J. Stewart. RFC2519: A Framework for Inter-Domain Route Aggregation, February 1999.

[19] Aman Shaikh and Albert Greenberg. Experience in black-box OSPF measurement. In *IMW '01 Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement*, pages 113–125, 2001.

[20] RFC2544 Latency Testing on the Arista 7100 Series. http://www.aristanetworks.com/en/products/7100/latencyreport.

[21] Pierre Francois, Clarence Filsfils, John Evans, and Olivier Bonaventure. Achieving sub-second IGP convergence in large IP networks. *ACM SIGCOMM Computer Communication Review*, 35:35–44, 2005.