Scalability Aspects of Centralized Control of MPLS Access/Aggregation Network

Dávid Jocha, András Kern, Kiran Yedavalli
Ericsson Research
{david.jocha, andras.kern, kiran.yedavalli}@ericsson.com

Abstract— **We apply the concept of centralized control to the MPLS access/aggregation network segment of service provider networks. We describe a numerical model to capture the scale and scope of the access/aggregation network and explain the corresponding scalability issues. We compare this numerical model to the experimental results obtained on a test-bed with OpenFlow 1.1 compatible central controller and forwarding elements. We show that the central controller implementation we have mostly matches the performance requirements of the access/aggregation networks for different scenarios.**

Keywords- OpenFlow; Scalability; numerical model; Onix; access/aggregation network

I. Introduction

Centralized control of networks using protocols such as OpenFlow [3] is gaining increasing momentum in the industry. This concept is an emerging area of research in multiple use-cases such as data center, enterprise, and service provider networks. Centralized control of networks an envisioned through OpenFlow has two central themes – separation and centralization of the control software from the forwarding elements in the network and a flow centric view of network traffic. This differs significantly from the main stream network architecture today, in which every forwarding element in the network has corresponding control plane software that runs distributed control algorithms with a packet centric view of the network. There are many advantages to the centralized control of networks - listing all of them is out of the scope of this paper – that include independent optimization of control and forwarding planes, higher service velocity in deploying new innovative services, and a more software centric view of the network.

In this paper we present our work on the application of the concept of centralized control to the access/aggregation segment of service provider networks. More specifically, we present our findings on the scalability aspects of the centralized controller controlling an MPLS access/aggregation network. In the current networks, MPLS is primarily restricted to the network core. In this work, we make use of the centralized control to extend MPLS all the way up to the access from the core of the network [7]. First we present a numerical model for the access/aggregation network to determine the size and scale of the network and network events for different scenarios. Then we present our experimental results based on a prototype of an OpenFlow-based controller and forwarding switches on a test bed and compare them with the numerical model.

The paper is organized as follows: We explain the access/aggregation use case in detail in Section II and present the numerical model in Section III. We present the results of our experiments in Section IV and conclude the paper in Section V.

II. Centralized Control of Access/Aggregation Networks

In state-of-the-art aggregation networks in use today, Ethernet-based aggregation is used for providing triple-play services (see BBF TR-101 [8] for a general overview) as well as dedicated business services, as defined by the Metro Ethernet Forum. The next generation of aggregation networks will extend this set of services with support for mobile backhaul, fixed mobile convergence (FMC), enhanced scalability and security mechanisms (see BBF TR-144 and BBF WT-145 (a.k.a. BBF TR-101bis) for additional details). To simplify network management and operation a unified MPLS-based solution is expected: it allows various connectivity abstractions (L1/TDM, L2/ATM/Ethernet/HDLC, L3/IP) suitable for a wide range of services, and to offer sophisticated OAM functionality. In addition to the introduction of MPLS in the aggregation domain, ideas for a joint, MPLS-enabled aggregation and core domain have emerged recently (e.g., IETF draft about Seamless MPLS [6]). Introducing an MPLS-based transport framework covering all network segments promises a separate evolution of transport and service architecture.

The migration to MPLS-based access/aggregation raises the question of how to organize the control plane. One possible solution is to deploy the IP/MPLS control plane in the access/aggregation as well [6]. However, this requires running IP/MPLS control stack on all access / aggregation nodes. This paper considers an alternative control plane solution, where the access/aggregation domain is provisioned by a centralized controller, which also interoperates with other controllers or the IP/MPLS control plane to establish end-to-end-connectivity.

A further gain of the centralized control is the possibility of the independent evolution of control and data plane.

A. *Overview of access aggregation networks*

Typically the access / aggregation part is organized in a hierarchical manner and consists of the following basic building blocks, assuming that services for residential and business customers as well as mobile backhaul customers are

This work was partially sponsored by the SPARC FP7 EU project: http://www.fp7-sparc.eu

organized via a common infrastructure (see Figure 1). Any customer equipment is connected to the Access Nodes (ANs) either through a Residential Gateway or directly. The traffic of these Access Nodes is aggregated by the aggregation network segment toward IP edges. The IP edges reside at the border of the aggregation and the backbone network. This latter network segment is typically an IP/MPLS network and is a meshed network former by Label Edge Routers (LERs) and Label Switching Routers (LSRs). The aggregation network is hierarchical by introducing two levels of aggregation switches (AGS1 and AGS2): the typical topologies here are trees, dual-homing and rings. A Service Edge may be connected to the network at any MPLS nodes of the network.

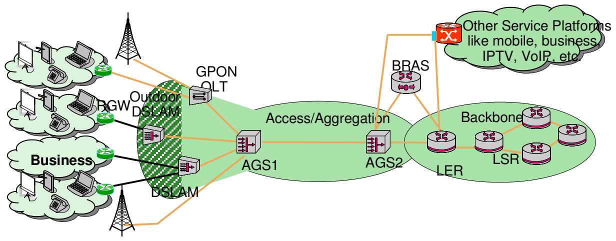

Figure 1: An access/aggregation network

B. *Scalability aspects*

Such MPLS based access / aggregation network should be scalable, which can be achieved: (i) If the available number space for addresses/labels/other protocol fields are enough for any large/complex structure we imagine; (ii) If reaction time and processing & network capacity requirements do not get unacceptably large for any large/complex structure we imagine; (iii) If the available number space of protocol fields, the reaction time and processing & network capacity requirements grow acceptably. Based on these requirements the following OpenFlow-related scalability concerns arise.

- There are theoretical constraints due to the protocol limitations such as maximum number of fields or objects, maximum length or value of fields/objects, maximum size of packets (MTU), and fragmentation. In our use-case the protocols to be considered are OpenFlow, OSPF, LDP and RSVP.
- The computational resources at the controller can be a bottleneck, since it's a single central entity in the network. The complexity of the used algorithms (like NP-hardness) and the time budget of running an algorithm (P could be still hard) are interesting issues.
- The control network's capacity limits the overall control traffic, introducing upper bounds on the frequency and size of node configuration commands. The network size (such as the number of nodes/ports/links) and reaction time is to be considered here.
- The control network introduces non-negligible delays due to geographical distance of switches and the controller. This gives a lower bound on reaction time even if the controller has infinite computational power.

In this paper we focus on selected requirements on the number of network devices and the controller performance.

III. Numerical Model

We created a numerical model to analyze some important static and dynamic aspects of configuring the access/aggregation network with a central controller via OpenFlow. This model is presented in this section. It should be noted that due to space constraints it is not possible to present all the input parameters in details in this paper.

A. *Model introduction*

In this section we present the details of the numerical model.

1) *Topology*

In line with the access/aggregation network shown in Figure 1 and according to the snowflake topology model [12], our OpenFlow domain consists of Access Nodes (AN), first level aggregation switches (AGS1), second level aggregation switches (AGS2) and edge nodes (EN). All other entities, both on the client and the core side are not part of the OpenFlow domain. Note that multiple access/aggregation domains can be connected to the Core network, however they have no direct interaction. The layout for such a topology is shown in Figure 2 below.

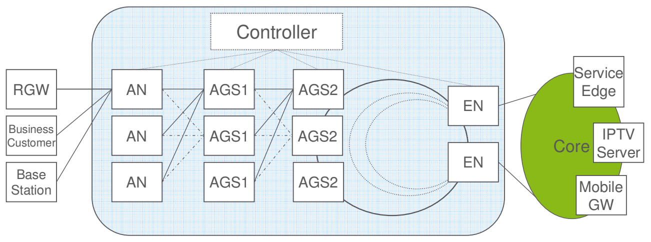

Figure 2: Topology Model

2) *Services*

The characteristics of services provided by the above network architecture, derived from the input to the European Union SPARC project [11], are as follows:

- There are Residential services, namely the Simple service for Internet access and the IPTV. The Simple service provides bidirectional connectivity between the RGW and the Service Edge. The IPTV service provides bidirectional connectivity between the RGW and the IPTV Server, where the direction from the IPTV server dominates, and typically the same data is simultaneously sent to multiple RGWs.
- There are Business services, namely the Simple service (like for residential customers), the Point-to-Point (PtP) and the Multipoint to Multipoint (MPtMP). The PtP service connects two Business customers, while the MPtMP connects multiple Business customers. All of these services are bidirectional.
- The Mobile Backhaul service connects a Base station to a Mobile GW in a bidirectional way.

3) *Transport connection set*

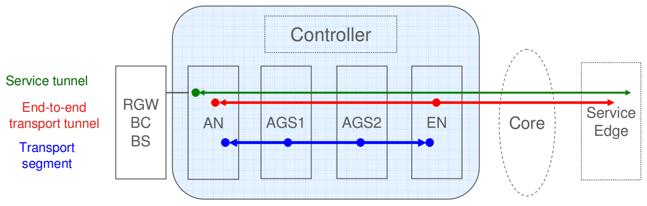

Figure 3: A possible connectivity model

To transport the above services appropriately, transport connection architecture is considered as depicted in Figure 3. The blue connections are OpenFlow domain internal transport segment tunnels. There are multiple options how to organize these tunnels. There can be a segment tunnel connecting each AN to the EN as seen in the figure. End-To-End tunnels (shown as red line) are defined between nodes, e.g. the AN and the SE. Such tunnels overlap OpenFlow and non-OpenFlow domains. Service tunnels (shown as blue) within the E2E tunnels identify the service, like the user in case of an AN where multiple RGWs are connected. This triple-layer connectivity set provides compatibility with the IP/MPLS core, flexibility within the OpenFlow domain, as well as it supports scalability (by aggregating connections).

4) Scenarios

Based on the Sparc project input [11] we have three scenarios, covering current and future deployment numbers, see Table I. For all scenarios, the relations between different network device numbers are as follows: customers devices (CE) >> customer edge (RGW) >> access nodes (AN) >> edge nodes (EN). Additionally, the sum of access nodes (AN) relates to the sum of aggregation (AGS1+ARS2) and edge node (EN) as 10:1. Based on these further assumptions, network topologies can be drawn for the different scenarios.

TABLE I.

Scenario	*Number of nodes in the domain*
Sparc today	~ 10 K
Sparc future	~ 20 K
Sparc long-term	~ 50 K

One further scenario we considered is based on the IETF Seamless MPLS draft [7], with an order of magnitude lower number of devices in a single access/aggregation domain - number of aggregation domains: 100, number of backbone nodes: 1000, number of aggregation nodes: 10000, number of access nodes: 100000.

B. Estimating the amount of configuration data

The total number of forwarding devices to be managed by the controller gives a requirement on the active network connections, which are tracked by the controller. For our scenarios, it will be in the order of magnitude of ten-thousands for the SPARC scenarios and thousands for the seamless MPLS scenario.

A further question of the scalability of the centralized control plane is the amount of configuration data, which must be handled by the data nodes or the controller. To estimate it with the number of flow entries to be deployed at different node classes as well as managed by the controller. Figure 4 shows the number of flow entries to be supported by the network entities for our access-aggregation use-case for different scenarios. The number of flow entries per forwarding device is in the magnitude of hundreds or thousands, except for the EN, for which it can go up to a few ten thousands.

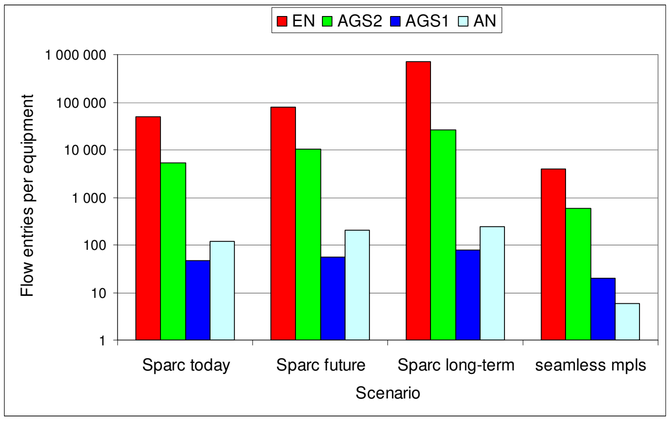

Figure 4: Number of flow entries

The overall number of flow entries to be managed by the controller in the domain is shown below in Table II.

TABLE II.

Scenario	*Controller flow entries*
Sparc today	~ 1.3 M
Sparc future	~ 4.4 M
Sparc long-term	~ 13.4 M
seamless mpls	~ 18 K

1) Set requirements for dynamic behaviour

In case of any unexpected network events, such as link failures, the control must be able to give proper answer within a predefined time constraints. The scalability aspects of such reactive operation can be estimated by the amount of configuration data to be sent as a response to a particular event. The more configuration data to be sent the faster controller will be able to fulfill the given time constraints.

Figure 5 shows the number of flow modification commands to be sent by the controller in case of restoration after a link failure. Best case and worst case refer to the possible minimal and maximal number of modification commands. (Minimal assumes using a direct backup link between the same nodes where a link failed, while maximal assumes rerouting the connection through a completely different path.) For the given specific scenarios, the results are in the magnitude of hundreds of configuration commands, except for the closest link to the Core, for which it goes to ten-thousands.

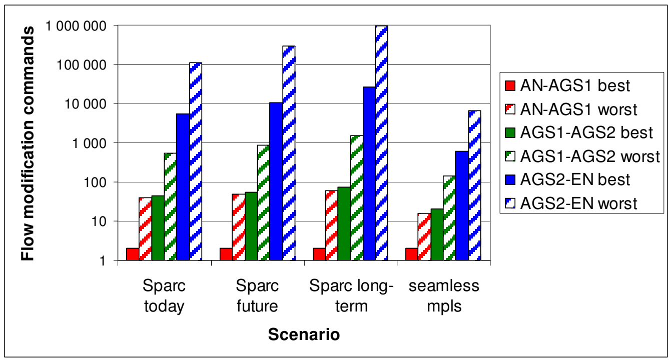

Figure 5: Flow mods in case of restoration

The high values close to the EN are the result of the many tunnels between the ANs and the EN. To decrease this high number of flow mod messages, those tunnels could be aggregated, to have only a single transport segment tunnel between neighboring nodes. In case of restoration, an alternative segment tunnel can be built between the same two nodes over multiple other nodes. This increases the static number of flow entries per forwarding device and the traffic volume; however, it decreases drastically the message number in the restoration case. Figure 6 shows the comparison of a tree (snowflake) topology, a tree + ring combination topology, and the same topology with the single hop transport segments.

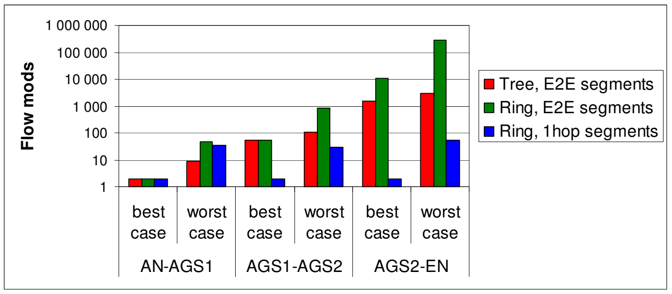

Figure 6: Tunnel options

IV. EXPERIMENTAL RESULTS

In this section we describe our tested for experiments and the corresponding results.

A. *Testbed*

For the central controller, we have used ONIX [5] middleware after extensive modification to make it OpenFlow 1.1 [3] compatible. We needed OpenFlow 1.1 as it supports MPLS natively compared to OpenFlow 1.0. Our controller hardware is an Intel Xeon 3GHz CPU with 24 cores and 16 GB RAM.

ONIX is a distributed network middleware that provides a component based mechanism to add applications on top of the middleware. It provides distributed storage mechanisms such as Distributed Hash Table (DHT) and SQL based transactional storage for network objects stored in the network information base (NIB), in addition to a distributed coordination mechanism based on Apache Zookeeper [9]. It also provides clustering mechanisms for a controller system made up of multiple servers. The clustering mechanism elects a master controller and all the other controllers in the cluster become slaves (stand-bys). All the switches in the underlying network connect to the master controller. ONIX uses a cooperative threading model implementation that limits the packet processing from ingress to egress to a single CPU core. Thus the performance of the controller depends on this aspect, and that it would improve if the cooperative threading mechanism is modified to become multi-core friendly.

For the forwarding plane we have used the Mininet network emulation tool [2] in conjunction with Ericsson developed open source OpenFlow 1.1 reference switch [10]. We have emulated up to 200 switches using Mininet connected to a single controller cluster. We consider a linear topology of switches in the forwarding plane in order to capture the worst case dynamics of the system. In the worst case, the length of the network would equal the length of the flow set up in the network.

We evaluate the controller performance for two different applications for varying network size – forwarding of packets and MPLS LSP creation. In the forwarding application, the forwarding switch encapsulates any hitherto unknown flows in an OpenFlow 1.1 packet-in message and sends it to the controller. A packet forwarding application on the controller handles the incoming packet-in and forwards the packet back to the switch without any modifications to it. This application tests the throughput of the application stack on the controller.

Similarly, for the MPLS LSP creation application, the switch sends the new, unknown flow to the controller and the application on the controller determines the Label Switched Path of the flow, and the OpenFlow 1.1 flow-mod messages that need to be created for all the switches in the flow-path and creates them searching through the NIB to obtain the information about adjacent switches for this purpose.

B. *Results*

Figure 7 below shows the performance of the controller for the two applications for varying network size. From the figure it can be deduced that the controller can be modeled as a simple queue that is served at a specific rate depending on the application. The saturation or the maximum throughput of the controller for a given application is obtained by saturating the queue with a packet incoming rate that is much higher than the serving rate of the controller application. According to the figure, the application serving rates for forwarding packets and MPLS LSP creation are 20000 packets/sec and 8000 flow-mods/sec respectively. In addition the maintenance of a forwarding element connection requires very little processing power on the controller side. It should be noted that these numbers are specific to the existing implementation of the ONIX controller that is based on a cooperative threading model and utilizes a single core of the CPU.

The above figure can be used to derive useful metrics such as maximum flow (e.g. LSP) creation rate given the length of the flow-path. For example, as shown in Figure 2, the maximum length of a LSP in the access/aggregation use case is 4. Thus for a LSP of 4 switches, the maximum LSP creation rate is 8000/4 = 2000 LSPs/sec. Thus, one LSP can be created in less than a milli-second in this case. It should be noted that the numbers here are limited to the controller performance. For example, the actual flow-path creation would include programming the flow-mod messages created by the controller into the switch which could be big component in the time required to program a flow in the network [6].

Comparing Figure 5 and Figure 7, clearly the current single-core implementation of the controller can support the restoration requirements for most of the scenarios within a second. However at higher aggregation layers of larger access/aggregation network we faced with scalability issues as it would take from ten seconds to two minutes to download the flow reconfiguration to the affected switches. Note that by introducing a protection sub-layer at the higher aggregation layer the configuration burst decreases, thus, the reconfiguration time will decrease to the order of hundred milliseconds.

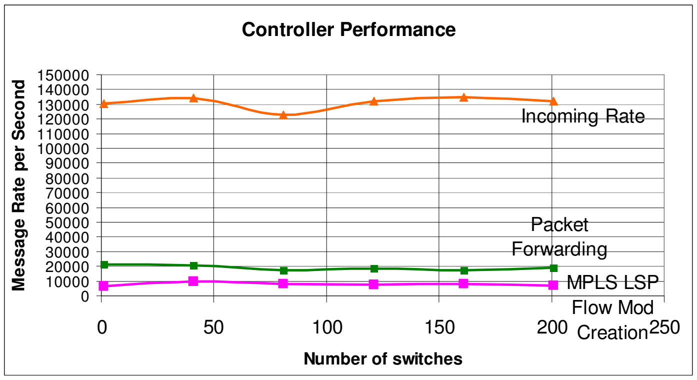

Figure 7: Controller Performance for simple forwarding and MPLS LSP creation applications

V. Conclusions and Future work

Our work looks for possible scalability limitations of applying an OpenFlow-based centralized control solution to the access/aggregation segments of the service provider networks. To understand and discover possible scalability constraints, first we formalized a numerical model. With this mode we estimated scale and performance numbers, which need to be matched by a centralized controller, considering various access/aggregation network sizes. Besides, with the help of this model we illustrated that how changing the tunnel structure can support decreasing the controller's reaction time in case of failure.

Further, we have discussed the experimental results obtained from running performance tests on a test-bed with OpenFlow 1.1 compliant ONIX-based central controller and forwarding elements. The current measurements reported that the performance of the current single CPU core implementation of the controller raised some scalability concerns in case of restoration – where strict time constraints are given. However, a multi-core implementation is expected to resolve these issues, which will be part of our future study.

Besides evolving the controller, we also will focus on further developing our numerical model placing more attention on the dynamics introduced by service creation and provisioning, like deploying IPTV.

References

[1] "ForCES Protocol Specification", IETF draft-ietf-forces-protocol-22

[2] "Mininet." [Online]. Available: http://yuba.stanford.edu/foswiki/bin/view/OpenFlow/MininetG

[3] "Openflow." [Online]. Available: http://www.openflow.org

[4] "Nox." [Online]. Available: http://noxrepo.org

[5] T. Koponen, M. Casado, N. Gude, J. Stribling, L. Poutievski, M. Zhu, R. Ramanathan, Y. Iwata, H. Inoue, T. Hama et al., "Onix: A distributed control platform for large-scale production networks," OSDI, Oct, 2010.

[6] Jeffrey C. Mogul, Jean Tourrilhes, Praveen Yalagandula, Puneet Sharma, Andrew R. Curtis, Sujata Banerjee, "DevoFlow: Cost-Effective Flow Management for High Performance Enterprise Networks"

[7] "Seamless MPLS Architecture", IETF draft-ietf-mpls-seamless-mpls-00

[8] Broadband Forum TR-101, "Migration to Ethernet-Based Broadband Aggregation"

[9] "Zookeeper" [Online]. Available: http://zookeeper.apache.org/

[10] "Ericsson OpenFlow 1.1 reference switch" [Online]. Available: https://github.com/TrafficLab/of11softswitch

[11] Sparc project, "D2.1 Initial Definition of use cases and carrier requirements", [Online] http://www.fp7-sparc.eu/project/deliverables

[12] IETF RFC 5439, "An Analysis of Scaling Issues in MPLS-TE Core Networks"

[13] Z. Cai, A. L. Cox, and T. S. E. Ng. Maestro: A System for Scalable OpenFlow Control. Tech. Rep. TR10-08, Rice University, 2010.