UNIVERSITY OF AMSTERDAM

MSC SYSTEMS & NETWORK ENGINEERING

Architecture of dynamic VPNs in OpenFlow

By:
Michiel APPELMAN
michiel.appelman@os3.nl

Supervisor:
Rudolf STRIJKERS
rudolf.strijkers@tno.nl

June 20, 2013

Summary

Contents

List of Figures

List of Tables

1 Introduction

Network operators today use Network Management Systems (NMSs) to get control over their devices and services that they deploy. These systems have been customized to their needs and in general perform their functionalities adequately. However, operators run into obstacles when trying to expand their business portfolio by adding new services. This will require *a*) new Application Programming Interface (API) calls to be implemented between their B/OSS and NMS, *b*) their NMS to be able to cope with potentially new protocols, and *c*) added expertise by engineers to define the requirements and restrictions of these protocols. When these obstacles are eventually overcome the setup that will result from this implementation will be relatively static, since any change to it will require the whole process to be repeated.

Until recently this limitation didn't distress operators as their networks were in fact primarily static. But with increasing demand for services requiring for example mobility and short-term virtual networks, these limitations start to become a tangible problem for operators. By solving the complexity of implementing new services or features for them, they will be able shorten their time to market, save on networking expertise and be more adaptive to changes in these services.

A potential candidate to solve this complexity is OpenFlow [1] and Software Defined Networkinging (SDN). SDN is a relatively new architecture to allow for the programmability of networks. The architecture has recently been standardized in the OpenDaylight project [2] which also includes OpenFlow, a lower level and increasingly supported API towards networking devices. Implementing the SDN architecture promises *a*) CAPEX savings due to hardware being more generic and flexible, *b*) OPEX savings because of the integration of NMSs and the control interface of the devices, thereby increasing automation, and *c*) increased network agility by using the open interfaces to program network devices directly [3].

Research Question

It is unclear however if a real-world OpenFlow and SDN implementation will actually provide any simplicity, additional flexibility or cost savings when compared to contemporary technologies [4]. Indeed, the technologies in use today have served operators well up until this point and their practicality has been proven over the past years. This research will seek to identify where exactly operators can benefit from implementing this use-case using SDN compared to the architecture in use today.

Doing so will help researchers and operators answer the question: *"How much can operators benefit from using OpenFlow when implementing dynamic VPNs (DVPNs)?"* This research offers that – with regard to the given use case – OpenFlow will reduce the complexity in the complete architecture of the management systems and network as a whole.

Scope

Given the use-case of implementing DVPNs similar to gTBN [5], this research will show the advantages of implementing such an environment using both Commercial Off-The-Shelf (COTS) technologies and an SDN solution. This will include research to determine if the proposed SDN and OpenFlow architecture will actually be able to support the creation of dynamic VPNs. The focus will primarily be on deploying Provider-provisioned VPNs (PPVPNs) at Layer 2 of the OSI-model between end-users. It's one of the more commonly used virtualization models, which is also plagued by its mostly manual and static setup procedure. When configured, users will be immediately connected to each other with minimal setup using a single broadcast domain.

Previous research in [6] has proposed a very specific implementation for programmable networks to deploy on-demand Virtual Private Networks (VPNs) but it predates the OpenFlow specification, and also omits a comparison with how this would look using contemporary technologies.

Should there be any time left, other use-cases to fill in this void will be composed in collaboration with the supervisor. These can include for example mobility within networks, multi-domain VPNs or smart metering of network usage.

Approach

We will start by giving a taxonomy of the state of the art in the networking industry with regards to different functionalities and features of the technologies. This overview will give an indication of where OpenFlow and SDN will be able to differentiate from contemporary protocols, be it in positive or negative. Using this data we will make an architectural design of the given use-case using both contemporary technologies and the SDN architecture and compare the effort in implementing such a design.

To quantify the efforts of both designs an overview will be made of the custom APIs and management systems will be made. The procedure to extend on the designs will also be defined, listing the required amount of changes to each architectural component and the expertise needed to implement these changes. Finally this will give an overview of the initial work and required work to adapt for both designs, and result in a recommendation for either one of the deployments.

2 State of the Art

The issue of creating DVPNs within Service Provider (SP) networks apparently comes from the inability to do so using technologies available to operators today. To get an understanding of how we got to this point and where the limitations or obstacles lie, an overview of the state of the art is required.

To be qualified to provide network operators with DVPNs each technology will need to be able to provide the following features:

1. scalable up to thousands of (dynamic) Layer 2 VPNs and client MACs,
2. fast failover times (<50ms) to provide continuity to critical applications,
3. efficient use of, and control over all network resources,
4. provide Quality of Service features to differentiate between classes of applications, and
5. an automated way to install VPNs in the network.

2.1 SDH/SONET

The foundation of Wide Area Networks (WANs) today is still Synchronous Digital Hierarchy (SDH) and its North-American counterpart Synchronous Optical Network (SONET). It operates at Layer 1 and is concerned with putting various data streams on fiber links. SDH networks find their origin in the telecommunications world where it was developed to transfer real-time cell switched voice data. It quickly evolved to include Asynchronous Transport Method (ATM) and later also Ethernet. Due to its maturity the protocol is considered stable but is also very static in its manageability. Developments are ongoing in the Generalized MPLS (GMPLS) field to bring dynamic configuration to SDH devices, but they are in their infancy.

2.2 ATM

ATM is a legacy protocol that has been used by operators to carry traffic over the internet backbone since the 1990s. Where as Ethernet and Internet Protocol (IP) are developed as packet-routing connection-less protocols, ATM is a cell-switched connection-oriented protocol. This poses a number of problems when trying to transport IP over ATM. First, the variable length of the packets don't map efficiently to the fixed size cells of ATM. Drops of a single cell would cause the entire frame to become unusable. Then there is the added overhead of encapsulating IP over ATM, which causes inefficient use of the network resources when compared to running an all IP network. Finally, the Quality of Service (QoS) features of ATM are left unused. These problems are some of the reasons that operators have moved away from ATM based backbones, to all IP ones.

As mentioned, ATM does provide QoS to the network operators and it also allows for the granular control of network paths – called circuits – but at a price. The Virtual Circuits (VCs) in an ATM network have to be tightly managed for the network to function properly. This limits its dynamism and scalability, which (also considering the aforementioned objections) does not make it a viable candidate to implement DVPNs.

2.3 MPLS

Multi Protocol Label Switching (MPLS) is known for its scalability and extensibility. Over the past decade additions have been made to the original specification to overcome a plethora of issues within carrier networks. This initially started with trying to implement fast forwarding in legacy switches using labels (or tags) at the start of the frame [7]. When this issue became surmountable using new hardware, MPLS had

already proven to be capable of transporting a wide arrange of protocols on the carrier backbone network, all the while also providing scalability, Traffic Engineering (TE) and QoS features to the operators.

MPLS itself is more a way of forwarding frames through the network, without facilitating any topology discovery, route determination, resource management, etc. These functions are left to a stack of other protocols. To discover the topology, MPLS relies on an Interior Gateway Protocol (IGP). The distribution of labels is done using Label Distribution Protocol (LDP) and/or Resource Reservation Protocol (RSVP), of which the latter also provides granular TE and QoS functionalities.

VPNs are also provided by additional protocols. Layer 3 VPNs make use of Border Gateway Protocol (BGP) to distribute client prefixes to the edges of the carrier network. The core is only concerned with the forwarding of labels and has now knowledge of these IP prefixes. Layer 2 VPNs make use of LDP and Virtual Private LAN Service (VPLS), a service which encapsulates the entire Ethernet frame and pushes a label to it to map it to a certain separated network. Again, the core is only concerned with the labels and only the edges need to know the clients MAC addresses.

Because of its extensibility the MPLS technology and the added protocols and tools, it is commonly used in Carrier Ethernet Networks (CENs) as an alternative to legacy ATM and SDH networks. With added features such as Equal Cost Multi Path (ECMP), Fast Reroute (FRR) and explicit routing it has proven to be a technology fit for carriers to transport critical applicition traffic over large networks. MPLS thus meets the requirements set forth, which might be the reason for its widespread use presently. However, given the scale and criticality of the networks and the stack of various protocols to implement it, management of MPLS networks has remained fairly static and mostly a manual task.

2.4 SPB

Shortest Path Bridging (SPB) is an evolution of the original IEEE 802.1Q Virtual LAN (VLAN) standard. VLAN tags have been in use in the networking world for a long time and provide decent separation in campus networks. However, when VLAN-tagging was done at the customer network, the carrier couldn't separate the traffic from different customers anymore. This resulted in 802.1Qad or Q-in-Q which added an S-VLAN tag to separate the client VLANs from the SP VLANs in the backbone. This was usable for the Metro Ethernet networks for awhile but when SPs started providing this services to more and more customers, their backbone switches could not keep up with the clients Media Access Control (MAC) addresses.

To solve this scalability problem Provider Backbone Bridging (PBB) (802.1Qay or MAC-in-MAC) was introduced. It encapsulates the whole Ethernet frame on the edge of the carrier network and forwards the frame based on the Backbone-MAC, Backbone-VLAN and the I-SID. The I-SID is a Service Instance Identifier, which with 24 bits is able to supply the carrier with 16 million separate networks. The downside of PBB remained one that is common to all Layer 2 forwarding protocols: the possibility of loops. Preventing them requires Spanning Tree Protocol (STP) which will disable links to get a loop-free network. Disadvantages of STP include the relatively long convergence time and inefficient use of resources due to the disabled links. This final problem was solved by using IS-IS as a routing protocol to distributed the topology and creating Shortest Path Trees (SPTs) originating from each edge device. This is called SPB or 802.1aq.

SPB benefits from the maturity of the Ethernet protocol by reusing protocols for Operations, Administration and Management (OAM) and Performance Measurement (PM) and the fact that only the edges of the network need to be SPB capable – the core switches just need to be able to forward 802.1Qad frames. Manageability of SPB VPNs is simplified to mapping a certain customer port or VLAN to an I-SID. Limitations of SPB include a very coarse way to apply TE policies or ECMP load sharing. Although drafts are in the works to resolve this limitation, this has yet to be standardized and implemented. Finally, the protocol at this point lacks fast failovers, and as such will not be a viable option to implement DVPNs.

2.5 TRILL

There has been discussion going on between SPB and Transparent Interconnection of Lots of Links (TRILL) supporters as to which is the 'better' protocol. Indeed, both use Intermediate System-Intermediate System (IS-IS) as an IGP and both try to solve the same problem to make Ethernet networks scalable to the desired scale of networks in use today, but they differ in their premise and implementation. TRILL adds a completely new header on top of the Ethernet frame with a source and destination RBridge allowing the so-called RBridges to route – Layer 3 – the frame to its destination over the TRILL backbone network.

Without this paper turning into another TRILL versus SPB comparison, it is important to point out various differences in the two technologies with regards to our proposed use-case. First, the hop-by-hop decision making by TRILL has the benefit of being able to distribute traffic in more efficient way over a dense network. SPB on the other hand makes its forwarding decision at the head-end, similar to MPLS forwarding. This means that careful consideration needs to take place to choose either more efficient ECMP forwarding or granular control over forwarding paths.

Additionally, because TRILL is a completely new protocol, the Internet Engineering Task Force (IETF) is also still working on adding some essential features that are still in draft. These include QoS, OAM, PM and TE. And finally there is another limitation for CENs, which is the lack of network separation. TRILL only supports the 4096 VLAN tags present in the 802.1Q frame to separate customer networks. At this point, it will fail to meet the requirements to provide DVPNs.

2.6 OpenFlow

Section 1 gave a short introduction into what SDN and OpenFlow entail and what it promises in terms of cost savings and agility. Software Defined Networkinging is the general principle of designing flexible networks using open interfaces towards the hardware. The complete architecture is being standardized by the networking industry within the OpenDaylight project [2]. Looking at OpenFlow itself from a more technical point of view, it boils down to a protocol used to program the forwarding plane in networking devices from a centralized server called the 'controller'.

The momentum that SDN is getting might be explained by a general need for change in the networking industry. Operators primarily want to get more control over their networks, something which using the current stack of protocols is relatively complicated to get. The original OSI reference model [8] touches on the "Management Aspects" of each layer in the model, a way for management entities in the highest layer to control the behavior of lower layers. Unfortunately in the swift evolution of Transmission Control Protocol (TCP)/IP, these management interfaces are often limited or absent all together.

The OpenFlow controller provides operators with an alternative to these interfaces, namely a programmable forwarding plane. This gives them TE and QoS control by using custom applications to direct traffic through the network.

The OpenFlow specification has been through a few revision since it was first proposed. In the first version tagging of traffic was only support using a single VLAN tag. Version 1.1 added matches for MPLS and Q-in-Q tags, and version 1.3 could also match PBB tags. These features are essential for implementing VPNs because the traffic flows will need to be separated upon entering the carrier network. Also, with scalability in mind we do not want to learn the client MACs in the core of the network.

Also not included in version 1.0 were mechanisms to allow for fast failover and ECMP. These were added in version 1.1 by introducing logical groups of ports to which a flow entry can forward the frame. However, to support fast failover a *liveness monitoring* technique will need to be implemented supported by the switch. This could simply be the physical link state or more intricate tools, e.g. Bidirectional Forward Detection (BFD).

3 Design

3.1 Requirements

1. What will the system need to look like?
2. What are its functional requirements?
3. What is the input?
4. What is the expected output?
5. How can it be extended in a later stage?

To further shape the actual design of the system, we will specify what this DVPN service will need to look like.

First, we will need to define a certain input. Our design accepts a set of physical ports with an optional C-VLAN corresponding to it. This set represents the group of users which will be placed in a single VPN and thereby connected to each other. It also requires the values for minimum and maximum bandwidth used by DVPN over the network. High bandwidth DVPNs may need load-sharing over multiple physical 1GbE links for example.

Second, the output will be defined as the VPN created throughout the network and Layer 2 connectivity between the chosen endpoints from the input.

Third, the usage of the network resources should be monitored during the lifetime of the DVPN. If certain paths are nearing their capacity, DVPNs should be able to be moved to paths where more resources are available.

Finally, the tearing down of the DVPN will also need to be arranged so as to free up resources for new DVPNs.

A design for the complete process will be given in the following sections, first in a situation where traditional network management and MPLS are used. And after that using the SDN approach and its way of managing network devices.

3.2 MPLS

To start:

1. Input goes where?
2. Flow of information.
3. Amount and type of output.

As has been discussed in Section 2.3 a typical MPLS network consists of a stack of routing protocols. This means that to provision the DVPN several protocols will be affected.

First, determine best path and configure RSVP to make paths between Provider Edges (PEs). Needs TE input.

Second, VPLS to add the ports to the VPLS instance and define the paths to use.

LDP monitored for adjacency between PEs.

In the access layer, MPLS has to be supported. Otherwise no mapping to customer VLAN, only per port. Other option: Q-in-Q but mapping of C AND S-TAG to VPLS instance needs to be supported by PE.

3.3 OpenFlow

To start:

1. Input goes where?
2. Flow of information.
3. Amount and type of output.

Need OpenFlow support on access-layer, or Q-in-Q to map to VPLS instance at PE, OR match on MAC.

4 Results

5 Conclusion

6 Future Work

A Acronyms

API Application Programming Interface

ATM Asynchronous Transport Method

BFD Bidirectional Forward Detection

BGP Border Gateway Protocol

CEN Carrier Ethernet Network

COTS Commercial Off-The-Shelf

DVPN dynamic VPN

ECMP Equal Cost Multi Path

FRR Fast Reroute

GMPLS Generalized MPLS

IEEE Institute of Electrical and Electronics Engineers

IETF Internet Engineering Task Force

IGP Interior Gateway Protocol

IP Internet Protocol

IS-IS Intermediate System-Intermediate System

LAN Local Area Network

LDP Label Distribution Protocol

MAC Media Access Control

MPLS Multi Protocol Label Switching

NMS Network Management System

OAM Operations, Administration and Management

OSI Open System Interconnect

PBB Provider Backbone Bridging

PE Provider Edge

PM Performance Measurement

PPVPN Provider-provisioned VPN

QoS Quality of Service

RSVP Resource Reservation Protocol

SDH Synchronous Digital Hierarchy

SDN Software Defined Networkinging

SONET Synchronous Optical Network

SPB Shortest Path Bridging

SPT Shortest Path Tree

SP Service Provider

STP Spanning Tree Protocol

TCP Transmission Control Protocol

TE Traffic Engineering

TRILL Transparent Interconnection of Lots of Links

VC Virtual Circuit

VLAN Virtual LAN

VPLS Virtual Private LAN Service

VPN Virtual Private Network

WAN Wide Area Network

B Bibliography

[1] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69–74, 2008.

[2] "OpenDaylight Project." http://www.opendaylight.org/.

[3] S. Das, G. Parulkar, N. McKeown, P. Singh, D. Getachew, and L. Ong, "Packet and circuit network convergence with openflow," in *Optical Fiber Communication (OFC), collocated National Fiber Optic Engineers Conference, 2010 Conference on (OFC/NFOEC)*, pp. 1–3, IEEE, 2010.

[4] J. Van der Merwe and C. Kalmanek, "Network programmability is the answer," in *Workshop on Programmable Routers for the Extensible Services of Tomorrow (PRESTO 2007), Princeton, NJ*, 2007.

[5] M. L. Cristea, R. J. Strijkers, D. Marchal, L. Gommans, C. de Laat, and R. J. Meijer, "Supporting communities in programmable grid networks: gTBN," in *Integrated Network Management, 2009. IM'09. IFIP/IEEE International Symposium on*, pp. 406–413, IEEE, 2009.

[6] B. Yousef, D. B. Hoang, and G. Rogers, "Network programmability for vpn overlay construction and bandwidth management," in *Active Networks*, pp. 114–125, Springer, 2007.

[7] Y. Rekhter, B. Davie, E. Rosen, G. Swallow, D. Farinacci, and D. Katz, "Tag switching architecture overview," *Proceedings of the IEEE*, vol. 85, no. 12, pp. 1973–1983, 1997.

[8] H. Zimmermann, "OSI reference model–The ISO model of architecture for open systems interconnection," *Communications, IEEE Transactions on*, vol. 28, no. 4, pp. 425–432, 1980.

Acknowledgements

Thanks to Rudolf Strijkers for his supervision during this project.