Hochschule für Technik und Wirtschaft Berlin
University of Applied Sciences

Studiengang Computer Engineering
Fachbereich 1
Prof. Dr. Thomas Baar

Design and implementation of a connectivity manager for virtual scalable network environments

Bachelorarbeit
von

Manuel Bergler

01.12.2014 – 08.02.2015

Referent: Prof. Dr. Thomas Baar
Korreferent: Prof. Dr. Thomas Baar
Betreuer: Benjamin Reichel M. Sc.

Manuel Bergler
Urbanstr. 26
10967 Berlin

Hiermit versichere ich, dass ich die von mir vorgelegte Arbeit selbstständig verfasst habe, dass ich die verwendeten Quellen, Internet-Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Berlin, den 08. Februar 2015

(Unterschrift)

_____

Manuel Bergler

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

1.1 Motivation

The demands on networks have changed dramatically in the past two decades, with an ever-growing number of people and devices relying on interconnected applications and services. The underlying infrastructure has been left mostly unchanged and is approaching its limits. In order to resolve this, Software Defined Networking (SDN) is going to be extending and replacing parts of traditional networking infrastructures. SDN separates the network into control and forwarding planes and therefore allows a more efficient orchestration and automation of network services.

The use of cloud-based services, with not only competitive pricing but also high-availability and fast network access, is taking over the traditional self-hosted data centers. The ease of administration and the deployment of new Virtual Machines (VMs) on the fly make it possible to create a Topology of Computers with no effort.

Network services have different requirements, depending on the type of data and their importance. The classification of network traffic can be done through Quality of Service (QoS). A new approach has to be made to enable the use of QoS in virtualized cloud infrastructures like OpenStack, to achieve controlled traffic from the deployment of Virtual Machines on.

1.2 Network Architecture

Today's traffic patterns, the rise of cloud computing and "big data" to only name a few examples, are exceeding the capacity of classic network architectures. With scalable computing and storage the common-place tree-structured network infrastructure with Ethernet switches are not efficient and manageable enough.

The increasing complexity of problems that have to be faced in networks and the need to control network traffic through software, are only a selection of the reasons why the Open Networking Foundation (ONF) developed an approach called Software-Defined Networking (SDN).

SDN is a leading-edge approach where the network control is separated from the forwarding

functions. The centralized network intelligence allows programming the network, without a need to access the underlying infrastructure. Therefore a shift of today's networks to more flexibility, programmability and scalability is going to take place.

1.3 Objective

The primary objective of this work is the development of a network orchestrator which is able to apply Quality of Service to the network interfaces of Virtual Machines. These Virtual Machines are deployed with OpenStack Nova and connected to an OpenVSwitch, which uses OpenFlow. Another task of the Connectivity Manager is to select which OpenStack hypervisor new VMs should be running on, which takes different runtime parameters into account. The CM should be able to be applied in environments with scalable hypervisors and VMs.

1.4 Scope

The scope of this work includes a Connectivity Manager which will have a Connectivity Manager Agent running on the cloud controller within the OpenStack infrastructure, to provide access to the hypervisors of OpenStack Nova. These two components have to be implemented and integrated with the existing OpenVSwitches. As a reference for a cloud infrastructure, multimedia communications like the Nubomedia project will be used. The deployment of this cloud is then tested on different performance characteristics like network bandwidth, latency, CPU utilization and memory usage.

In virtualized cloud infrastructure like OpenStack, the placement of Virtual Machines (VMs) on a particular compute node can be decided on by comparing different run-time parameters. The network connectivity between those VMs has to be prioritized and classified into different classes, depending on the service that are running on it.

Currently there are a number of solutions for managing network connectivity between VMs. A comparison and their current limitations follows in the next section. The chosen approach is to extend the existing network control and management services with Quality of Service (QoS) capabilities. In support of the thesis the Connectivity Manager will be implemented and the differences in bandwidth usage will be shown in one use-case.

1.5 Overview

Chapter 1 begins with the motivation for this thesis and gives a brief introduction into the objectives and the scope.

Chapter 2 gives an overview of traditional network concepts and a introduction to SDN and its components. Furthermore the different services that make up OpenStack will be described.

Chapter 3 conceptualizes the state-of-the-art solutions that are currently available and evaluates their implementation and limitations.

Chapter 4 contains an analysis of requirements and an architectural overview of the Connectivity Manager. Moreover design aspects are introduced and illustrated according to their requirements.

Chapter 5 examines the implementation of the Connectivity Manager and Agent.

Chapter 6 evaluates the network performance tests on the basis of a particular use-case.

Chapter 7 summarizes the results of this work and gives an overview on possible future work.

Chapter 2

Fundamentals and Related Work

2.1 Networking

As a basis for further explanations in regards to network concepts of the design and implementation, this section describes the essentials of network technology.

2.1.1 Fundamental network characteristics

Firstly the traditional network technologies that are used for inter-networking are explained.

IP based communication

TCP/IP model IPv4 TCP / UDP traffic protocols

Switches and routers

Quality of Service

2.1.2 SDN architectures

In this section the concept of Software-Defined Networking is explained.

Software-defined network concept

Characteristics & Benefits

SDN architecture

OpenFlow Controller

OpenFlow Switch

OpenFlow Software Switch

OpenVSwitch

2.2 Cloud computing infrastructures

2.2.1 OpenStack

2.2.2 Devstack

2.2.3 OpenStack Nova

2.2.4 OpenStack Heat

2.2.5 OpenStack Neutron

2.3 Conclusion

Chapter 3

State of the art

3.1 Requirements

3.1.1 Functional requirements

3.1.2 Non-functional requirements

3.2 Existing S-O-T-A solutions

3.2.1 OpenDaylight SDN controller

3.2.2 Ryu SDN controller

3.2.3 OpenStack Neutron - QoS Extension

Chapter 4

Design

4.1 Architecture overview

4.2 Design of Connectivity Manager

4.3 Design of Connectivity Manager Agent

4.4 Conclusion

Chapter 5

Implementation

5.1 Environment

Software development approach

Project structure

5.2 Connectivity Manager components and operations

Selection of best-matching Hypervisor

Enabling QoS for VM

5.3 OpenStack Neutron configuration

5.4 Conclusion

Chapter 6

Evaluation

6.1 Feature analysis

6.2 Nubomedia use-case

6.3 Conclusion

Chapter 7

Conclusion

7.1 Summary

7.2 Problems encountered

7.3 Future work

Appendix A

List of source codes

Anmerkungen: Quelltextauszüge zu einer Implementierung sind im Anhang dann sinnvoll, wenn einige, spezielle Implementierungstechniken aufgezeigt werden sollen, die in der Darstellung als Algorithmus oder Pseudocode nicht deutlich werden. Keinesfalls soll der gesamte Quelltext angehängt werden und weiterhin soll auch in einer Vorbemerkung die Auswahl der Quelltextauszüge genau erklärt werden.

Verwendet wird das freie Quelltext-Pretty-Printing-Tool a2ps.exe mit folgender Aufrufkonvention (vgl. [a2ps 07, grep 07]):

```
a2ps.exe --pretty-print=cxx -i test.cpp -o test.ps -T3
```

Die Quelltextdateien dürfen hierfür eine Zeilenlänge von 80 nicht überschreiten. Die störenden Kommentare im Header und Footer des entstandenen .ps-Files können mithilfe des freien Tools grep.exe automatisiert entfernt werden. Eine Batch-Datei für den gesamten Konvertierungsprozess inklusive Konvertierung in das pdf-Format hat beispielsweise folgenden Inhalt:

```
a2ps --pretty-print=cxx -i %1 -o tmp -T3
grep -v "Gedruckt von" tmp | grep -v ") footer" > %1.ps
del tmp
"c:\programme\adobe\acrobat 7.0\Acrobat\acrobat.exe" %1.ps
```

Appendix B

Glossar

Anmerkung: Das vorliegende Glossar wurde ohne die Zuhilfenahme der speziellen Glossarumgebungen von Latex erstellt, um eine etwas freiere Formatierung nutzen zu können.

2,5D-Datensatz →Tiefenbild.

3D-Modell, 3D-Modellerfassung (optische) Der Begriff des 3D-Modells wird in der vorliegenden Arbeit für wasserdichte Oberflächenmodelle verwendet. Dies dient zur Abgrenzung gegenüber 3D-Volumenmodellen und →Tiefenbildern. Der Begriff der Optischen 3D-Modellerfassung umschließt hier neben der eigentlichen Sensordatenauswertung auch die →3D-Registrierung und die Schritte der Nachbearbeitung wie Glätten und Ausdünnen der Daten und Stiching-Operationen.

3D-Registrierung Vgl. Abschnitte 2.4, 4.3 und →Registrierung.

Bibliography

[Abdel-Aziz 71] Y. I. Abdel-Aziz and H. M. Karara, „Direct linear transformation from comparator coordinates into object space coordinates in close-range photogrammetry“, in: Symposium on Close-Range Photogrammetry, issue 11, pp. 1–18, University of Illinois at Urbana-Champaign, 1971.

[AutTech 07] Firma Automation Technology GmbH in 22946 Trittau, Produktübersicht, Downloads und Datenblätter. URL: `http://www.automationtechnology.de`

[Dang 06] T. Dang, C. Stiller and C. Hoffmann, „Self-calibration for Active Automotive Stereo Vision“, Proc. of the IEEE Intelligent Vehicles Symposium, pp. 364–369, Japan, Tokyo, 2006.