Title Placeholder

Subtitle Placeholer

Networks and Distributed System

Group 1025 - 10$^{\text{th}}$ semester

Aalborg University
Department of Electronic Systems
Fredrik Bajers Vej 7B
DK-9220 Aalborg

Department of Electronic Systems
Fredrik Bajers Vej 7
DK-9220 Aalborg Ø
http://es.aau.dk

AALBORG UNIVERSITY
STUDENT REPORT

Title:
Title Placeholder
- *Subtitle Placeholer*

Theme:
Master's thesis

Project Period:
10^{th} semester
Networks and Distributed System
February – June, 2014

Project Group:
1025

Participants:
Martí Boada Navarro

Supervisors:
Jimmy Jessen Nielsen
Andrea Fabio Cattoni

Copies: 3

Page Numbers: 15

Date of Completion: April 1, 2014

Abstract:

Nowadays networks use different kind of mechanisms in order to give priority to a certain sort of traffic to process the packets in a different way depending on their application. The most common way to achieve this is using Differentiated Service, marking the packets depending on their application (giving higher preference to those that are more important or time sensitive such as Voice over IP, or Video on Demand). However, determining resource allocation per class of service must be done with knowledge about traffic demands for the various traffic classes, keeping a fixed amount of bandwidth for each class, which results in a poor utilization of resources.

In the last years, a new networking approach called Software-Defined Networking (SDN) is emerging fast. This approach is based on the separation of data and control planes. Such approach allows the network administrator to have a more dynamic control of the network behaviour. The purpose of this project is to analyse the possibilities that SDN provides to develop a more efficient resources allocation along the network.

Preface

This is a report of a student project conducted on the 4^{th} semester of Networks and Distributed Systems, at Aalborg University. This project serves as a Master Thesis as the requirements of the master's programme dictate. This is applied in the comparison of a designed application used for Cognitive Networks concepts over a Software-Defined Network (SDN).

In this project Floodlight is used as an SDN controller, and the network is simulated using mininet software.

Aalborg University, April 1, 2014

Martí Boada Navarro
<mboada14@student.aau.dk>

Contents

1 Introduction

This introduction chapter tries to give a general view of the existing problems in the current networks, and how new emergent technologies can help to improve the network possibilities and capabilities in order to achieve a more flexible approach and adapt to todays needs.

1.1 Motivation

In the last 20 years networks requirements have been changing constantly, the amount of traffic have been increasing exponentially and more demanding end-to-end goals are needed. However, the networks architectures have been unchanged, increasing the complexity and hindering its configuration. In order to adapt to the new needs, new network paradigms such as Software-Defined Networking (SDN), cognitive networks or automatic networks are emerging fast due the interests of carriers and ISPs.

The first things to analyse are the problems of the existing networks. They have become a barrier to creating new, innovative services, and an even larger barrier to the continued growth of the Internet.

Analysing the data plane in the ongoing networks architecture we find well defined layers for different purposes, making them autonomous. For instance the physical layer (fibre, copper, radio...) is independent of the link layer (ATM, X.25, PPP...), the transport layer (TCP, UDP...) or the application layer (HTTP, FTP, telnet...), what allows the evolution of each of them independently. Thanks to the fact of dividing the problem in tractable pieces, networks have been able to evolve, increasing many magnitude changes in terms of speed, scale or diversity of uses.

On the other hand, there is the control plane which, unlike the data plane, has no abstraction layers at all. Within the control plane there are several protocols to configure the network elements, such as Multiprotocol Label Switching (MPLS)[1] or NETCONF[2]. There are also a bunch of routing protocols e.g. Routing Information Protocol (RIP)[3], Enhanced Interior Gateway Routing Protocol (EIGRP)[4] or Shortest Path Bridging (SPB)[5].

[1]MPLS architecture RFC 3031
[2]Network Configuration Protocol (NETCONF) RFC 6241
[3]RIP Version 2 RFC 2453
[4]CISCO Enhanced Interior Gateway Routing Protocol (EIGRP)
[5]SPB IEEE Std 802.1aq

Protocols tend to be defined in isolation, however, with each solving a specific problem and without the benefit of any fundamental abstractions. This has resulted in one of the primary limitations of today's networks: complexity. For example, to add or move any device, IT must touch multiple switches, routers, firewalls, Web authentication portals, etc. and update ACLs, VLANs, Quality of Services (QoS), and other protocol-based mechanisms using device-level management tools. In addition, network topology, vendor switch model, and software version all must be taken into account. Due to this complexity, today's networks are relatively static as IT seeks to minimise the risk of service disruption.

In order to deal with all that mess, hundreds of network monitoring and management tools[6] have appeared trying to help the networks managers to have a control of their networks.

At the same time that technology advances, the communications requirements also evolve. The claims that nowadays are needed in the different services that users demand, such as VoIP or streaming video in High Quality, where inconceivable when the architecture was designed.

All those factors have lead to an over-provisioning of the networks, increasing the cost, and wasting resources due the difficulty to optimise the control and adapt the physical resources available to the requirements of every instant. So it is time to take a step forward and evolve to a more optimal architecture, easy to manage, evolve and understand.

Here is where the SDN approach (explained in section 2.3) comes to play. Having a clear abstraction layers and a centralised control plane, it's much easier to make a more efficient and dynamic management and control of the network, since we have a global overview of the network, and control over its entirety.

The motivation of this Master's Thesis is to take advantage of the new centralised networking approach of SDN to dynamically allocate the appropriated resources (appropriated QoS) to each sort of service or user.

1.2 Problem formulation

Data traffic in networks have been increasing exponentially since the beginning of networking at the same time that new kind of services and connection requirements appear. Those factors have led to a more complex networks that cannot satisfy the needs of carriers nor users.

As explained in this introduction chapter, the current networks have several limitations to adapt to end-to-end goals requirements for the different sort of services. Determining resource allocation per class of service must be done with knowledge about traffic demands for the various traffic classes, keeping a fixed amount of bandwidth for each class, which results in a poor utilisation of resources. Nowadays methodologies are static, or need specific network equipment, which leads to a non-scalable and expensive systems. QoS is

[6] Standford Network Monitoring and Management Tools list.

not a one-time deployment, but an ongoing, essential part of network design. Is because of that that SDN can improve significantly the management of the QoS performance, in a much easier way than the current networks.

The aim of this project is to analyse the possibilities that Software-Defined Networking bring us to develop an application able to sense the state of the network and adapt its behaviour in order to achieve a better performance and better resource utilisation of the network.

Thus, this project tries to answer the following questions:

> *How much can SDN improve the resources allocation in relation with current mechanisms?*

> *Is it easy to implement a module in SDN that adapts the network behaviour to the present conditions guaranteeing a QoS in a scalable way?*

2 Pre-analysis

In order to investigate how to answer the initial problem stated in section 1.2, some background information is needed. This chapters gives a general idea about the concept of QoS, analysing the current mechanisms used to achieve it. It also explains the traffic characterisation basis of the different sorts of traffic in which the project is focused, and also presents a general overview of the SDN architecture.

2.1 Quality of Service (QoS)

Quality of Service (QoS) refers to prioritise a specific sort of traffic in front of another. This is an important factor in order to adapt to different traffic models, and to provide a good Quality of Experience (QoE) to the end-user.

Fundamentally, QoS enables you to provide better service to certain flows. This is done by either raising the priority of a flow or limiting the priority of another flow. When using congestion-management tools, you try to raise the priority of a flow by queuing and servicing queues in different ways. The queue management tool used for congestion avoidance raises priority by dropping lower-priority flows before higher-priority flows. Policing and shaping provide priority to a flow by limiting the throughput of other flows. Link efficiency tools limit large flows to show a preference for small flows. There are three main parameters to determine what is the QoS requirements for each service:

Delay: it manifests itself in a number of ways, including the time taken to establish a particular service from the initial user request and the time to receive specific information once the service is established. Delay has a very direct impact on user satisfaction depending on the application, and includes delays in the terminal, network, and any servers.

Jitter: delay variation is generally included as a performance parameter since it is very important at the transport layer in packetised data systems due to the inherent variability in arrival times of individual packets. However, services that are highly intolerant of delay variation (such as voice) will usually take steps to remove (or at least significantly reduce) the delay variation by means of buffering, effectively eliminating delay variation as perceived at the user level (although at the expense of adding additional fixed delay).

Loss packets: has a very direct effect on the quality of service finally presented to the user, whether it be voice, image, video or data. In this context, information loss is not limited to the effects of bit errors or packet loss during transmission, but also includes the effects of any degradation introduced by media coding for more efficient transmission.

Nowadays networking QoS can be divided in 3 classes, Best Effort (BE), Guaranteed Services (IntServ) and Differentiated Services (DiffServ and MPLS-TE). Each of this approaches have their own positive and negative things. This sections give a general overview of each in order to understand the problems and the current methods to achieve end-to-end goals, dealing with the different kind of packets.

Table 2.1 summarises the features of this three types of QoS, specifying the service, service scope, complexity and scalability of each one.

Table 2.1: QoS comparison

	Best-Effort	Guaranteed	Differentiated
Service	Connectivity No isolation No guarantees	Per flow isolation, Per flow guarantee	Per aggregation isolation, Per aggregation guarantee
Service Scope	End-to-end	End-to-end	Domain
Complexity	No setup	Per flow setup	Long term setup
Scalability	Highly scalable, nodes maintain only routing state	Not scalable (each router maintains per flow state)	Scalable (edge routers maintains per aggregate state and core routers per class state)

Figure 2.1 represents the three levels of QoS mentioned above.

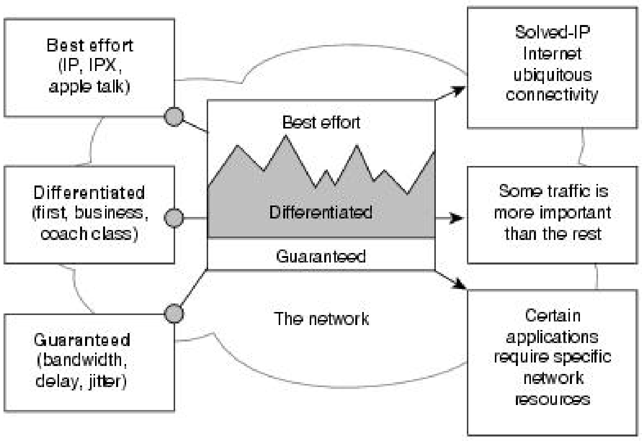

Figure 2.1: QoS levels representation

2.1.1 Best Effort

Best Effort is the current way the Internet is working, and is equal to do nothing with the packets, meaning that the service provided depends on the actual state of the network.

The positive points about BE is that is highly scalable, since the nodes just maintain routing state, and there is no need of any set-up. On the other hand this approach has no assurances about delivery, no control access, no isolation and no guarantees.

2.1.2 Guaranteed Services

Guaranteed Services guarantees specific resources for a specific flow, which means that can guarantee a QoS for the flow.The most popular mechanism of guaranteed services is the Integrated Services (IntServ), which uses the Resource Reservation Protocol (RSVP)[1] as a signalling protocol, which is the responsible of sending specific messages to the network nodes to reserve the required resources per each data stream. RSVP declares the QoS requirements and characterise the traffic of the the flow.

IntServ allows to differentiate three kinds of services: Guaranteed (real-time applications), Controlled load (applications that can adapt to network conditions within a certain performance window) and Best effort.

Even though IntServ ensures the specific QoS required per each flow, it has large scalability problems because maintaining states by routers in high speed networks is difficult due to the very large number of flows., besides all the routers have to be RSVP compatible.

2.1.3 Differential Services

In a traditional Internet Protocol (IP) networks each router performs an IP lookup for each packet to determine the next-hop based on its own routing table, and forwards the packet to that router. Each router makes its own independent routing decisions, until the final destination is reached. Differentiated services avoids the independent routing decision moving that computational cost to the edges of the network.

There are two main mechanisms to develop such kind of behaviour: DiffServ and MPLS-TE.

[1] Resource ReSerVation Protocol (RSVP) RFC 2205.

DiffServ

Due the limitations of IntServ (2.1.2), DiffServ appeared solving some of the problems. DiffServ consist on marking the packets with a priority stamp (Differentiated Service Code Point or DSCP) on the edge routers, and the core networks use this stamp to know the forwarding priority. This implies adding a labelling time, but since it is done in the edge routers, where the links speed are slower, it doesn't represent a problem.

The advantage of DiffServ is that scalable, since it doesn't require the routers to maintain state information for each flow, which is a huge burden for the routers. However, it also has problems. One is that since the packets are marked just at the edge routers, it can not solve the congestion inside the domain, so it cannot provide per flow bandwidth and delay guarantees. For example, a lot of flows in the same class can be routed through the same link, thus cause congestion there.

Another weakness of DiffServ is the lack of granularity for QoS guaranteed services, which makes it difficult to cost-effectively support end-to-end QoS according to the end-to-end situation (e.g., path lengths) of applications. With the conventional packet-level QoS mechanisms for the regulated traffic, i.e., buffer admission control plus output schedulers in general, increasing service granularity may inevitably complicate implementation and/or impact scalability since sophisticated output schedulers seem necessary in this case.

MPLS Traffic Engineering (MPLS-TE)

Multi-Protocol Label Switching (MPLS)[2] does a label switching instead. That means that the first router does a routing lookup but instead of finding a next-hop, it finds the final destination router and applies a label (or "shim") based on this information. The next routers will check this label to route the packet without needing to perform any additional IP lookups. The idea was to have only the first router doing an IP lookup, then all future routes in the network could do switching matching based on a label, reducing the load on the core routers, where high-performance was the most difficult to achieve, and distribute the routing lookups across edge routers.

MPLS-TE takes advantage of the MPLS labels in order to provide an efficient way of forwarding traffic throughout the network, avoiding over-utilised links, adapting to changing bandwidth taking in account the configured bandwidth of the links.

[2] MPLS architecture RFC 3031

MPLS-TE DiffServ Aware (DS-TE)

MPLS-TE Differential Services Aware (DS-TE) is a MPLS-TE able to detect the DCSP labels.

2.2 Traffic characterisation

In order to provide the appropriate quality for each service, a traffic characterisation is needed. This section gives a global overview of the requirements of three different sorts of traffic: voice, video and web-browser.

The International Telecommunication Union, Telecommunication Standardisation Sector (ITU-T) has a recommendation for "End-user multimedia QoS categories"[1] which is shown in Figure 2.2.

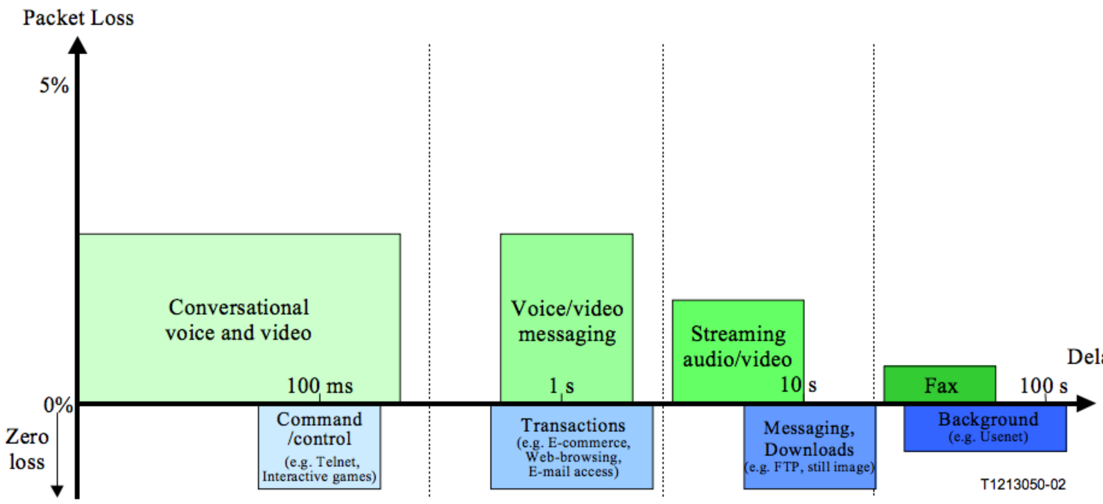

Figure 2.2: QoS requirements (ITU-T Rec. G.1010)

Nonetheless, this document is old, and due the constant evolution of services, it is still studied by ITU-T Study Group 12 (Performance, QoS and QoE)[3].

Although the conditions of the voice and voice traffic have no changed significantly, the video requirements are much higher. For instance, in the mentioned document, the ITU-T describes the typical data rate of the videophone service between 16 and 384 Kbps, but if we check the requirements of Googles videophone service[4], they suggest 1 Mbps for video-calls, and they mention that the minimum required is 256 Kbps outbound and 512 Kbps inbound.

[3] ITU-T Study Group 12 - Performance, QoS and QoE
[4] Google Hangouts system requirements

Table 2.2 shows the QoS requirements for each of the 3 services mentioned above.

Table 2.2: Performance targets for voice, video and web-browser applications

		Video		
	Voice	One-way	Two-way	Web-browser
Loss %	3	1	1	0
Jitter (ms)	10	10	10	NA
Delay (ms)	150	10000	150	10000
Typical datarates	64 Kbps	512 Kbps	512 Kbps	NA

2.3 Software-Defined Networking

One of the tendencies that seems to have more power for the next generation networking is Software-Defined Networking (SDN). One of the reasons to believe that SDN will be *the one* is that some big companies, such as Google, are already using it.[3][2].

SDN is a new network architecture that allows be programmed as if it were a computer. It provides an abstraction of the forwarding function decoupling the data plane from control plane, which gives freedom to manage different topologies, protocols without many restrictions from the physical layer.

As it is explained in section 1.1, the main problem of the current networks is that the control plane has no abstractions. That means that there is no modularity, which leads to limited functionality, since protocols with different proposals (such as routing, isolation or traffic engineering) coexist in the same layer. SDN was designed to change that paradigm, dividing the control plane in three main layers: the forwarding model, the Network Operating System (NOS) and the control program.

Forwarding model: composed by the Network Elements (i.e. switches) and a dedicated communicated channel from each NE with the NOS which uses a standard way of defining the forwarding state.

Network Operation System: piece of software running in servers (controllers) which provides information about the current state of the network such as the topology or the state of each port.

Control program: express the operator goals, and compute the forwarding state of each NE to accomplish those goals.

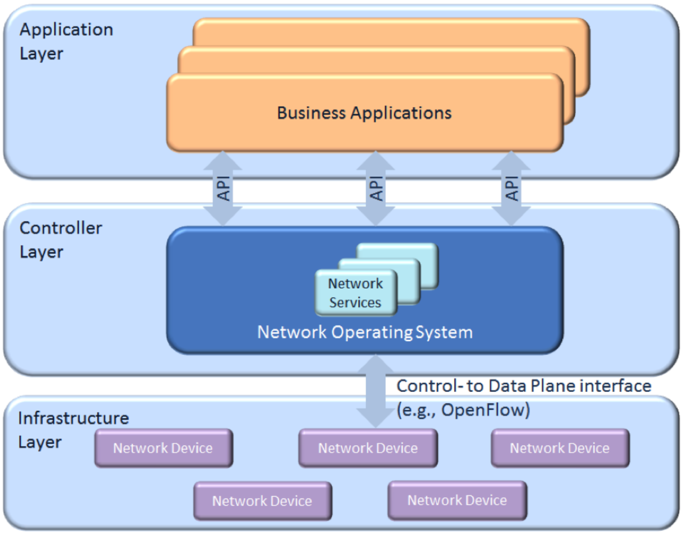

Figure 2.3: SDN architecture (image from Open Networking Foundation)

2.3.1 SDN advantages

SDN tries to improve the current networks. Bellow there is a list of the main advantages of the SDN paradigm.

OPEX reduction: centralised control helps to eliminate manual interaction with the hardware, improving the uptime of the network.

CAPEX reduction: separating the data plane of the control plane brings to a more simple hardware and increases the possibility of more competence between hardware manufacturers, since the devices don't depends on the proprietary software.

Agility: since the control layer can interact constantly with the infrastructure layer, the behaviour of the network can adapt fast to changes like failures or new traffic patterns.

Flexibility: having a separated abstraction for the control program allows to express different operator goals, adapting to a specific objectives. Operators can implement features in software they control, rather than having to wait for a vendor to add it in their proprietary products.

2.3.2 SDN applications

To give an idea of how huge SDN is, the list below mentioned some of the applications which is related to.

1. **Appliance Virtualization**
    - Firewalls, Load balancers, Content distribution, Gateways
2. **Service Assurance**

- Content-specific traffic routing for optimal QoE, Congestion control based on network conditions, Dynamic policy-based traffic engineering

3. **Service Differentiation**
   - Value-add service features, Bandwidth-on-demand features, BYOD across multiple networks, Service insertion/changing
4. **Service Velocity**
   - Virtual edge, Distributed app testing environments, Application development workflows
5. **'Traditional' Control Plane**
   - Network discovery, Path computation, Optimisation & maintenance, Protection & restoration
6. **Network Virtualization**
   - Virtual network control on shared infrastructure, Multi-tenant network automation & API
7. **Application Enhancement**
   - Specific SDN application, Reserved bandwidth for application needs, Geo-distributed applications, Intelligent network responses to app needs.

2.3.3 OpenFlow

OpenFlow is the protocol used to communicate the NOS (controller) with all the Network Elements (NE). Is an open standard that provides a standardised hook to allow researchers to run experiments, without requiring vendors to expose the internal workings of their network devices. OpenFlow is currently being implemented by major vendors, with OpenFlow-enabled switches now commercially available.

OpenFlow is the most common protocol used in SDN networks, and is often confused with the SDN concept itself, but they are different things. While SDN is the architecture dividing the layers, OpenFlow is just a protocol proposed to convey the messages from the control layer to the network elements.

Bibliography

[1] (2001). End-user multimedia qos categories. Technical Report ITU-T Rec. G.1010, International Telecommunication Union.

[2] (2012). Inter-datacenter wan with centralized te using sdn and openflow. White paper, Google, Inc.

[3] Sushant Jain, Alok Kumar, S. M. (2013). B4: Experience with a globally-deployed software defined wan. Technical report, Google Inc.

A Project proposal