Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Dept. of Information Security and Communication Technology

Sissel-Johanne Aspdal

Optimized use of CDN technologies in Uninett

Master's thesis in Communication Technology
Supervisor: Otto Wittner
June 2020

NTNU
Norwegian University of
Science and Technology

Sissel-Johanne Aspdal

Optimized use of CDN technologies in Uninett

Master's thesis in Communication Technology
Supervisor: Otto Wittner
June 2020

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Dept. of Information Security and Communication Technology

Title:	Optimized use of CDN technologies in Uninett
Student:	Sissel-Johanne Aspdal

Problem description:

According to statistics from Cisco more than 50% of all Internet traffic goes to/from Content Delivery Network nodes, and predictions say this will increase to more than 70% in the next couple of years. Hence ISPs need to consider how CDN traffic is to be handled in their network. Uninett, the national research IP network provider in Norway, is such an ISP. They need to better understand the CDN traffic distribution in their core network, and to examine if CDN traffic can be managed more optimally than today's scheme.

The project has two main objectives. First, measurements of CDN traffic in Uninett's core IP network will be planned and performed. This will include finding a general method of how CDN traffic can be distinguished from other traffic. Second, based on analysis of the discovered CDN traffic combined with network topological details, models will be developed for more optimal CDN traffic management. Attempts will be made to suggest novel and efficient schemes for handling the traffic applying current and/or futuristic CDN technologies, e.g. like Information Centric Network (ICN).

Responsible professor:	Otto Wittner, Uninett
Supervisor:	Håvard Eidnes, Uninett

Abstract

Measuring the amount of Content Delivery Network (CDN) traffic, and using CDN technologies to optimize content delivery in an ISP, have been the main topics of this master thesis. The global amount of CDN traffic is predicted by Cisco to be larger than the non-CDN traffic, which makes content delivery important for ISPs. Using CDN technologies to achieve an efficient handling of the content delivery is highly relevant.

Uninett is a Norwegian ISP, and provides the national research and education network. Uninett is used as a use case through this project to test the measurement methods and to develop a model how CDN technologies can be used to optimize handling of content delivery.

The measurement method is developed with using simple and available tools. The analysis suite SiLK is used together with Autonomous System Numbers (ASN) associated with CDN providers to achieve results. A set of model criteria have been created, where the two most important are to postpone upgrading central inter-domain links, and handling flash crowds. It was also important that the recommended model could be realistic to implement short-term.

The recommended model includes the solutions implementing CDN nodes in the ISP network and establishing peering agreements between the ISP and CDN providers. These solutions both contribute to moving the traffic load consisting of content away from central links, and postpone the need of upgrading the network. The amount of CDN traffic per CDN provider is the foundation of the specific recommendations to Uninett.

Sammendrag

Hovedteamet i denne masteroppgaven er målinger av trafikk forbundet med innholdsdistribusjonsnettverk (CDN[1]) og bruk av CDN teknologier for å optimalisere innholdsdistribusjonen for en nettverksleverandør. Cisco hevder at den globale mengden CDN-trafikk er større enn ikke CDN-trafikk, noe som gjør distribuering av innhold viktig for en nettverksleverandør. CDN teknologier kan brukes til å effektivisere håndteringen av innholdsdistribusjon, og er veldig relevant.

Uninett er en norsk nettverksleverandør, og er ansvarlig for det nasjonale forsknings- og undervisningsnettverket. Uninett er brukt som et eksempelstudie gjennom dette prosjektet for å teste målemetoder og for å utvikle en modell for hvordan CDN teknologier kan brukes for å optimalisere håndteringen av innholdsdistribusjon.

Målemetoden er utviklet ved å bruke tilgjengelige verktøy. Verktøyet SiLK brukes sammen med autonomt systemnummer (ASN), som tilhører CDN aktører, for å oppnå resultater. Et sett med kriterier til modellen har blitt utformet, og de to viktigste er at modellen skal bidra til at oppgradering av sentrale utenlandslinker utsettes, og modellen skal kunne håndtere unormale trafikkmønster med økt trafikk (flash crowds). Det har også vært viktig at den anbefalte modellen skal være realistisk å implementere på kort sikt.

Den anbefalte modellen inkluderer to løsninger, implementering av CDN noder i nettverket til nettverksleverandøren, og etablering av samtrafikksavtaler mellom CDN aktører og nettverksleverandører. Disse løsningene bidrar til å flytte innholds-trafikken vekk fra de sentrale utenlandslinkene, og utsetter behovet for oppgradering av linker. Mengden CDN trafikk per CDN aktør danner grunnlaget for de spesifikke anbefalingene gitt til Uninett.

[1]Content Delivery Network

Preface

This master thesis consummates the Master of Science (MSc) degree in Communication Technology at the Norwegian University of Science and Technology (NTNU).

The project topic was given by Uninett, and was chosen due to my interest in topics related to computer networking. I knew when I started that CDN was a wide topic and that it definitely would give me a real challenge. It did, and I have increased my knowledge and have taken even more interest into the area. This master thesis has given me a lot of insights how the networking world works in real-life, and communicating with different CDN providers has been very interesting. Thanks to all who have provided me with answers!

I would like to thank my responsible professor Otto Wittner and my supervisor Håvard Eidnes for providing excellent support through this process, and for making this master thesis manageable. One special thanks goes to Arne Øslebø for executing all my SiLK scripts and providing me with the needed results to complete this master thesis. I have to thank Angela Horneman at CERT for providing great advice how to use SiLK to accomplish my goals. Lastly, appreciations also goes to the Uninett employees that took part in the expert-group to validate my work.

Sissel-Johanne Aspdal
Trondheim, Norway
June, 2020

Contents

List of Figures

List of Tables

List of Acronyms

AANP	Akamai Accelerated Network Partner.
AI	Artificial Intelligence.
APNIC	Asia Pacific Network Information Centre.
ASN	Autonomous System Number.
CaTE	Content-aware Traffic Engineering.
CCN	Content Centric Networking.
CDN	Content Delivery Network.
CERT NetSA	CERT Network Situational Awareness Team.
GDP	Gross Domestic Product.
GGC	Google Global Cache.
HD	High Definition.
ICN	Information Centric Networking.
IPFIX	IP Flow Information Export.
ISP	Internet Service Provider.
IXP	Internet Exchange Point.
NDN	Named Data Networking.
NetPaaS	Network Platform as a Service.
NFV	Network Function Virtualization.
NIX	Norwegian Inernet Exchange.

NREN	National Research and Education Network.
OCA	Open Connect Appliances.
PNI	Private Network Interconnect.
POP	Points of Presence.
QoS	Quality of Service.
RTT	Round-Trip Time.
SCP	Smart Control Plane.
SD	Standard Definition.
SDN	Software Defined Networking.
SFI	Settlement-Free Interconnection.
SiLK	System for Internet-Level Knowledge.
UHD	Ultra-High Definition.
VNI	Visual Networking Index.
WWW	World Wide Web.

Chapter 1
Introduction

The amount of traffic transported via the Internet is increasing, and has been from the early start with ARPANET around 1970 [CAoCS18]. Numbers from Cisco states the traffic in 1984 to be 15 GB per month [Sum15], and a report from 2018 predicted it to be 254 EB per month in 2020 [Cis18]. Cisco further predict it to keep increasing. The amount of connected devices and the percentage of the population who have access to the Internet follows the same trend, and is increasing [Cis18].

The way Internet services are used has changed since the beginning. In the beginning communication through email was important, and later the World Wide Web was invented and much more information became public and possible to share with a wider audience [CAoCS18].

Content is King

Bill Gates
1996

As far back as in 1996 Bill Gates wrote an essay with the title ”Content is King” [Bai10]. This was on-point, and more and more content became available. Today video streaming is one of the most important Internet services [Cis18]. Video in itself requires a certain bitrate, but with the increasing qualities available, it requires even more bitrate. The development in video quality from Standard Definition (SD) to High Definition (HD) to Ultra-High Definition (UHD), has increased the needed bitrate with a factor of four (from HD) and nine (from SD) [Cis18]. This together with the increase in Internet users, and the number of devices, is contributing into increasing the traffic load in today’s Internet.

The world spanning network, the Internet, connects computer networks together. The physical infrastructure in the Internet transport network is not easily changed. One example can be communication links, like e.g. fibre optical cables, which connect the computer networks together. Such cables can be installed over huge distances, and even as submarine cables. The cost for installing long cables can be several hundred million dollars, and is time-consuming. These cables have limited capacity and life-span, and new ones must be added when all the capacity has been used or after a certain amount of time. This is one of the reasons why alternative technologies to handle the increasing traffic, have been developed. Alternative technologies can contribute to decreasing the core load. This can at least delay the need of new infrastructure, and still contribute to a well-functioning network and satisfied users.

In addition to the traffic increasing, users expects more and more when it comes to quality, delay etc. It is not enough to deliver the traffic, it has to be done in an efficient way, or it can lead to dissatisfied users [BPV08].

Video is one type of content that users are interested in today, and it is important to deliver the content as efficient as possible. One way to achieve that, is using technologies like Content Delivery Network (CDN). The concept of CDN is to move the content closer to the users. This can be done e.g. by having a set of servers with the same content distributed to distinct geographic locations. Even though the original content is placed at a server in New York in the US, a user in Norway can get the content from a server somewhere in Europe. With the use of this technology, the used network capacity is at the edge (from the server to the user), not the core of the Internet transport network. Cisco reports the amount of CDN traffic to be 56% in 2017, and forecast it to be around 65% in 2020, and 72% in 2022 [Cis18]. With such high amounts, it is important for an Internet Service Provider (ISP) to be aware of the possibilities, and to use CDN technologies to provide an efficient and cost-efficient network.

The topic of this project is to measure and analyse CDN traffic, and use the results to create a model how CDN technologies may be used by an ISP. Uninett is a Norwegian ISP that provides the National Research and Education Network (NREN), and will be used as a case study. In addition, future technologies like ICN will be addressed.

1.1 Motivation

The reasons for using CDN-technologies are several, and some of them motivate this project. For each reason, Uninett is applied as a use case to illustrate relevance and realism.

User experience may be improved by using CDN-technologies. If a user have to wait for the requested content, e.g. a lot of buffering when streaming a video, or experiencing degraded quality, they may loose interest and skip watching the video

[DSA+11]. This is an undesirable situation for all parts. Uninett is a research and education network, and one of Uninett's most important tasks is to facilitate for the education and research. They describe themselves as the ”digital foundation” [Uninda] in Norwegian education and research. To be able to serve the users, and best possibly support education and research, it should be efficient to get hold of needed content. In addition, the user experience of the general network performance is important. Using CDN-technologies can contribute to these goals.

CDN-technologies move content closer to the users, which can decrease the load in the global core network. Upgrading the infrastructure may be postponed. This applies to Uninett's infrastructure as well if they use CDN technologies. This can affect, and possibly decrease, the load on inter-domain links in Uninett. In best case, upgrading network infrastructure, like e.g. optical components like SFPs, can be postponed. Then the capacity of the links can be withheld for a longer time, which can be beneficial economically.

Traffic engineering and good knowledge about the traffic in the network, is central to an ISP. Being aware of the amount of CDN traffic, and which CDN providers that are present, will make it easier to optimize the network. The knowledge of CDN traffic in Uninett is more limited than desirable. By using Uninett as a case study, they will increase the knowledge, and may use it to optimize their network. It will also give an indication if statistics from Cisco is relevant for Uninett, as a NREN.

In addition, future technologies, like Information Centric Networking (ICN) may be of interest to an ISP to better handle the CDN traffic, and to optimize the network even more.

1.2 Thesis Scope

The scope of this report is to look into CDN in relation to an ISP. As mentioned earlier in this chapter, Uninett is used as a case study and represents the ISP. CDN-technologies are already used by Uninett, but this project will look into how they can benefit from using the technologies and if the solution used today is a good approach. To be able to find a answer to this, the amount of CDN traffic in Uninett will be mapped. This includes developing a method to measure CDN traffic, which is a significant part of this project. The available tool is SiLK, and is used to filter out the CDN traffic. The results from the measurements are analysed, to gain a better understanding of how the use of CDN technologies can be improved. Especially, by focusing on which CDN providers that are most present in Uninett's network. With the help of experts at Uninett, CDN providers and literature, possible solutions to optimize the use of CDN technologies are looked at. This will end in a model of how CDN-technologies can be used in Uninett, but the findings may be of use for similar

ISPs. To make the project even more relevant, some thoughts on future technologies, like ICN, will be briefly discussed.

1.3 Objectives

The objective and research questions are almost the same as the ones presented in the project preparing for this master thesis [Asp19]. The main objective is to map the amount of CDN traffic in Uninett, which will lead to a recommendation how CDN technology best can be used.

How can an ISP, like Uninett, most efficiently utilize content delivery technologies in their network?

To cover this objective, some research questions can be defined:

- How much of the traffic in Uninett is CDN traffic?
- Which CDN providers are most present in Uninett?
- How can the tool SiLK be used to map CDN traffic?
- Which criteria should the recommended model be based on?
- Which solutions can contribute to optimizing the use of CDN technologies?
- Can Information Centric Networking (ICN) contribute to a more efficient network?

1.4 Limitations

The project will not evaluate how the recommended model impacts the intra-domain links in the ISP's network. The inter-domain links is considered as the most expensive links to upgrade, and therefore they are the focus in this project.

Performance is not looked into, besides that CDN in general improve performance. The measurement method does not include this, and therefore the recommended model will not be based on performance measures.

1.5 Contributions

The contribution of this project is to provide a simple method how CDN traffic can be mapped with the use of the analysis suite SiLK. This approach can be reused, and adapted into similar ISPs as Uninett. Further, the findings are used to look into relevant solutions which is well-known and easy to implement to a network. The findings in this project may be used to achieve quick results with regard to link capacity savings and postponing costly upgrades of central links.

1.6 Thesis Outline

The master thesis is structured as follows:

- **Chapter 1** is an introduction to the project, containing the motivation, scope, objectives, limitations, and the contributions of this project
- **Chapter 2** provides background and relevant theory to be able to understand the work done in this project
- **Chapter 3** describes how the CDN traffic was measured, and presents the results
- **Chapter 4** presents the recommended model for optimized handling of CDN traffic in an ISP, and the different aspects that was considered to create the model
- **Chapter 5** discuss the findings in this project
- **Chapter 6** concludes this master thesis' work, and suggests further work

Chapter 2 Background and Theory

The following chapter presents relevant background and theory. Several topics will be covered to establish better preconditions for reading this project. First, some history is presented to better understand why CDN emerged and why it is so popular. Then basic CDN will be explained. Different types of CDN providers will be listed, and examples will be given. Uninett will be presented, together with how they use CDN technologies. One section will include how ISPs and CDN providers can cooperate, which is highly relevant. Which tools that were considered is presented briefly. Lastly, ICN will be looked into.

First of all, the term content is heavily used throughout this project. Content may be almost any types of contents available through the Internet, and Akamai have described content as following [Akandd]:

> ”4K and HD-quality video; audio streams; software downloads such as apps, games, and OS updates; data records that contain medical and financial information; and much more. Potentially any data that can be digitized can be delivered through a CDN.”

2.1 CDN History and development

To understand why the CDN technology was developed, some history is presented to point out different challenges with the Internet, and how CDN responds to those challenges. General advantages with CDNs are presented in the end. A basic presentation of what CDN is will be presented in Section 2.2, and can be read before this Section if needed.

As mentioned in Section 1, ARPANET was developed in the late 1960s [CAoCS18], and the early use of the network was mostly related to communication. Before ARPANET, it was normal to communicate by phone and physical mail. With the invention of ARPANET, e.g. services like e-mail was made available. In the beginning the network was used mostly by academia, but it quickly evolved to be available for commercial use, and the possibility to share content (e.g. files) became an important area of usage [CAoCS18].

Around 1990 the network of networks; the Internet, was a fact. More and more computer networks were connected together, and the development continued into a world-spanning network. Next, the World Wide Web (WWW) became public available in 1991 [CAoCS18], and made information and content accessible to a larger audience. Since the early start, the Internet has expanded in nearly every way possible. The amount of traffic, the numbers of users, the number of devices, broadband speeds etc., are some parameters that have increased, and which keep increasing [Cis18]. Users have embraced the development and possibilities, and the way users use and what they expect from Internet services, has changed through time. What users find interesting, is one such change. Users are interested in content, and video is the dominant traffic type in the Internet [Cis18]. In addition, the expected quality has increased [DSA^{+}11]. All this increasing parameters, together with the shift in user interest and expectation, does not scale well with the original design of the Internet transport network. Some new thinking, how the huge amount of content should be delivered, was required.

Another challenge was the cost and way to bill for Internet connectivity. In the early days, not everyone was convinced that Internet was going to become very popular. Therefore, billing systems was adopted from the phone business, and payment was done per minute [Rob17]. This was not necessary a fair and appropriate way to do the billing when the traffic is packets, and not one customer using a circuit. A new way to bill based on the amount of traffic, was not seen as an easy solution. Especially, when the traffic travelled through third party networks, which none of the parts exchanging traffic, were directly connected to. For smaller ISPs or content providers it existed some methods how they could buy capacity in the backbone Internet, with either fixed lines with a predefined capacity, or an on-demand method paying for the amount of MB per month. Increased traffic would either increase the price, or the service would be degraded if the link capacity was too low [Rob17]. It was hard to scale the network to the increased use of Internet services in a cost-efficient way, and alternative solutions like CDN appeared.

The first CDNs emerged back in the late 1990s, but the development was slow. The following paragraph is a reuse of some of the background from the project preparing for this master thesis [Asp19], describing the reason for the slow development.

The development of CDN has been slow, and some of the reason for that is due to patent. The history begins with Acacia buying patents for audio and video transmissions with some kind of server in between the client and the content owner.

This happened in the mid 1990s. They did not want to share this technology, and was more interested in earning money from the patent, than developing the technology. The patent was invalidated so late as in 2009. Further on, the big CDN service provider Akamai has done similar things regarding patents, for example the conflict with Limelight which was resolved so late as in 2016. All these patent laws tie up resources, and have most likely slowed down the development of CDN [Rob17].

The issues with patent also shows that one of the important goals for CDN providers was economical. Delivering content in the most cost-efficient way is a goal for every player in the CDN ecosystem.

CDN can contribute to solving challenges like increased traffic, billing, cost efficient delivery and still deliver high quality content with high performance to the users. The following paragraphs will mention some of the reasons why CDN technology appeared, why it became so popular and in general why using CDN is an advantage. The reasons presented is economical, performance and scaling.

Economical. The cost of operating an ISP or provide content is one of the reasons why CDN appeared [Rob17]. By using CDN, content providers do not have to lease their own capacity to deliver their content world-wide. They can buy CDN services from a CDN provider who already has servers around he world and other necessary infrastructure. ISPs may delay upgrading their network, especially when it comes to the expensive inter-domain links in their network, by using CDN technologies. Agreements between ISPs and CDN providers can even be ”free of charge”, and collaboration can be a beneficial situation for both the ISP and CDN provider [FPS+13]. One benefit for the CDN provider is to make their service more available for users, which can make them a more attractive CDN provider for the content providers. The following paragraph is reuse of some of the background from the project preparing for this master thesis [Asp19]. It further describes why economical reasons was important in the development of CDN technologies:

How fast traffic travels through the network is decided by the slowest link of the path, often called the bottleneck. Earlier, enterprises based their network architecture on a model that said that 80% of the traffic was LAN-traffic, and the remaining 20% was WAN-traffic. This is not the case today, and a lot more is WAN-traffic. The links from LAN to WAN (Internet edge) are not scaled for this, and often becomes the bottleneck. This is called the bottleneck theory [Cisnd]. The cost of upgrading these links is not necessary beneficial, and a better solution can be CDN. In mid to late 1990s, forward proxies were used to address this issue. This was less costly than leasing link capacity. So economic reasons is an important part of why CDN emerged [Rob17].

Performance. By using CDN technologies bottlenecks can be avoided. Users can get content from near locations, which can lead to reduced latency [NSS10]. Latency in networking can be defined as the time it takes to send a request from a sender to a receiver, and for the receiver to process this request. CDN providers may have deployed several servers around the world, close to the user, which makes the path between the users and the content shorter. The users will then experience less latency than if they had to fetch the content at the origin server at the other side of the world. With users requiring better and better service, lower latency is a reason to use CDNs [BPV08].

Scaling. Using CDN technologies makes it easier for content providers to scale their service. One example may be if a content provider is used to having a number of users using their service, and suddenly the usage increases heavily. This is an abnormal situation that can be called flash crowd [AJ00]. The content provider can solve this by scaling the service for very high usage, but this is not cost-efficient. CDN is a better option. In addition, without using CDN technologies, the content provider may not be able to meet the increased amount of requests. If this causes bad reception quality, it can end in users stopping to use the content from the content provider [PB12]. By having several extra servers with the same content, the computational power also increases, and a lot more requests can be handled compared to only using the origin server [Ver02].

The mentioned reasons are important for why CDN technologies are so popular. In the Cisco Visual Networking Index (VNI) from December 2018 [Cis18], they forecast the amount of CDN traffic to go keep increasing over the next years, and that it is bigger than the amount of non-CDN traffic. Figure 2.1 shows the forecast.

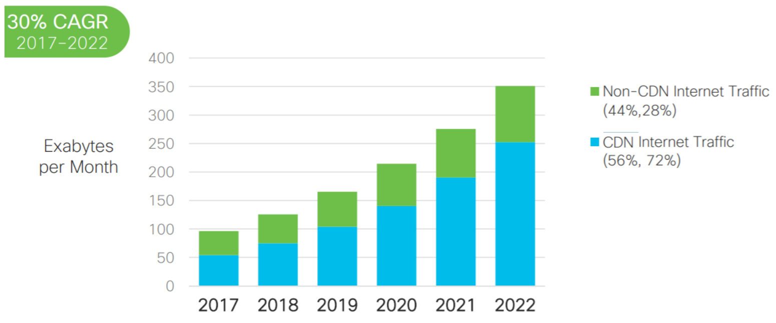


Figure 2.1: Amount of global CDN traffic-forecast [Cis18]

2.2 Basic CDN

The main idea of CDNs is to make the content from an origin server available at several replica servers or cache servers spread out geographically around the world

[BPV08]. The area of CDN consists of a big variation of solutions, and describing every aspect is not relevant. The following will describe the basic idea of CDN. In this project the actors in a CDN architecture is defined to be the content provider, CDN provider, the ISP and the users. This is mainly based on [BPV08], with ISP as a complement to the original actors. The content provider holds the origin content, which is stored at the origin server. The users are interested in this content, and the CDN provider makes the content available around the world with good performance. To do this the CDN provider has deployed several servers at different geographic locations, which holds copies of the original content. The servers are often located at the network edge, close to the users. These servers can go by several names, e.g. surrogates, edge servers and replica servers. In this project they will be referred to as edge servers. Figure 2.2 illustrates basic CDN where the content provider has an origin server which is placed in USA, the CDN provider places edge servers in different Points of Presence (POP) around the world, users get the content from an edge server and ISPs transport the content between the edge servers and users.

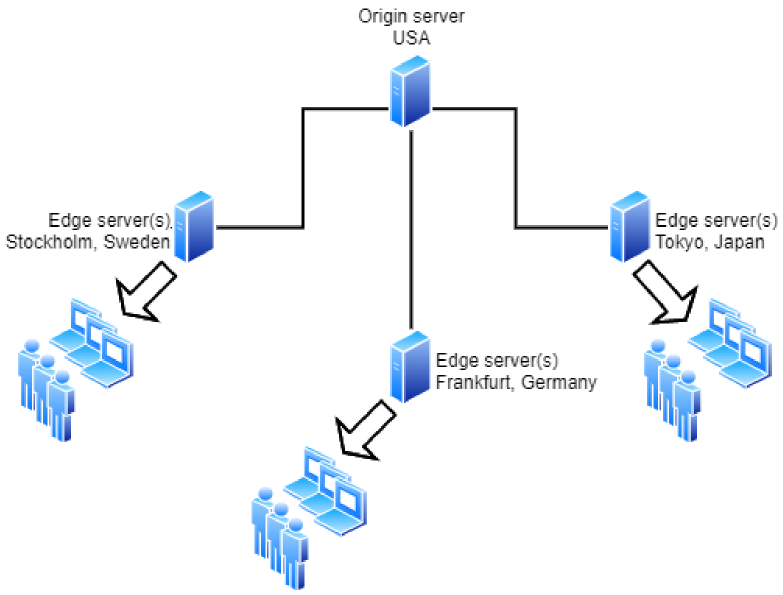


Figure 2.2: Illustration of CDN

A CDN consists of different parts, and is described in [BPV08]. The different parts can be put into the main categories of content delivery, request routing, distribution and accounting. The content delivery part is the origin and edge servers, the request routing part makes sure the users get served in the best possible way from an edge server, the distribution part is how content gets distributed from the origin server to the edge servers and lastly, the accounting part is to keep track of the traffic so that billing can be done correctly. All the CDN parts are needed to deliver content to the users in the most efficient way, and at the same time keep track of the CDN business. Figure 2.3 illustrates the parts.

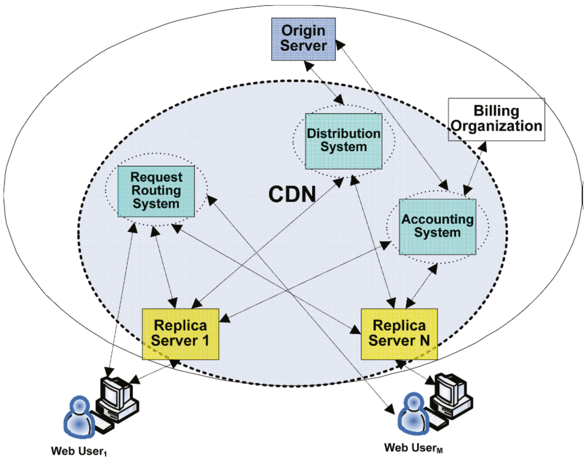


Figure 2.3: Illustration of CDN parts [BPV08]

Which edge server that is serving the user is decided by the request routing system. This is transparent to the user. This process of selecting server is one of the main challenges in CDN technology [FPS+13], and is often referred to as the server selection problem. Parameters that can be used by the request routing system is e.g. distance (which should be as low as possible), where the network congestion is smallest, and by avoiding edge servers with high load [CAR18]. The request routing system can be implemented in several ways, and two overview methods will be mentioned. When a user requests some content by typing an URL into the web browser, the request is sent to the content provider. The content provider may redirect the request to the CDN provider, and then the CDN provider choose which edge server that should answer the request [PB07]. Traditionally DNS and HTTP are used to solve the request routing.
Another method that can be used is IP anycast. By using this method, several edge server may be configured with the same IP-address, and the routing system decides which server that will serve the user request [CGF+15].

2.3 CDN providers

CDN providers can be categorized into different types, and the following list is based on definitions in the book ”Advanced Content Delivery, Streaming, and Cloud

Services” [PSR14].

– Pure-Play CDN

– Carrier/Telco CDN

– Managed CDN

– Licensed CDN

– Federated CDN

Out of the five types listed, it is the first two which often is referred to in the literature. Pure-play CDN is also often called commercial CDNs. Managed CDN refers to when commercial CDN providers help to operate CDN services in an ISP network. Licensed CDN is when e.g. ISPs buy CDN solutions from the commercial CDN providers, but operate it themselves. Federated CDN is when several CDN providers connect with each other. In addition to the listed types, private CDN, content CDN, cloud CDN, academic CDN and P2P CDN may be referred to as types. The types that will be used in this project are the same as used in Kentik’s customer portal [Kennd]. Kentik is a network analysis tool, and their portal have been accessible through NORDUnet. Kentik uses the following types: commercial, content, cloud and telco [Kennd]. A CDN provider may be a combination of several types, and is not restricted to one type. One example is Google, which provide services which may define the type as commercial, content and cloud. In the end of this section, examples of one commercial and one mixed type will be given.

Commercial CDN providers are typically providers which have been in the market for a long time, and originally were pure CDN providers [BS16]. They are often present all around the world, and content providers buy their services. Examples of such providers are Akamai and Limelight.

Content CDN providers are spreading their own content using CDN technology. Content CDN may also be private CDNs. Private CDNs have their own geographic network infrastructure, which can be leased capacity from network operators [CCL18]. In this case the edge server is owned by the content CDN provider, and they host their own content [pri17]. Some providers even build their own capacity, which e.g. providers like Google [goo19], Facebook and Microsoft [MA16] have done. Content CDN provider examples are Google and Facebook.

Cloud CDN providers build their CDN based on cloud storage providers. The infrastructure is consisting of data centres with storage possibilities and delivery capabilities, and is built on virtualization [HWC$^+$16]. This also opens up for smaller

Table 2.1: Overview of the CDN provider types

Type	Characteristic	Examples
Commercial	Sell CDN services to content providers	Akamai, Limelight
Content	Uses CDN technology to distribute their own content	Facebook, Google
Cloud	Uses cloud-technology to build a CDN	Microsoft Azure, Amazon
Telco	Own their own telecommunication infrastructure	Level3, Telenor

CDN providers, which can reach all over the world without having expensive data centres, by renting cloud capacity. They do not have to own the infrastructure, which can make them a relevant provider even though they are not a big commercial CDN provider [CGLLP12]. Big known cloud CDN providers are Microsoft Azure and Amazon, which are companies that own cloud infrastructure.

Telco CDN providers is telecommunication companies that provide CDN solutions [DVR11]. Telecommunication companies can also be referred to as ISPs. These operators have a slightly different view than commercial CDN providers, due to that they want to optimize the underlying network and infrastructure, in addition to the overlying services [LS13]. Examples may be Level 3 and Telenor.

Hybrid solutions exist, where e.g. content CDN providers mix up with commercial CDN providers. More specifically, this means that e.g. some of the edge servers that relay the content of a content provider may be owned by the content providers themselves, and some may be bought from commercial CDN providers [pri17].

Akamai is an example of a commmercial CDN provider. They were the first CDN provider, and started their services in the late 1990s [Akandc]. Then the operation was mostly focused on pure CDN services, but the operation has evolved, and it is now said that only 50% of the revenue is from the CDN part [BS16]. Akamai has been tested to be the best performing CDN provider in Norway [Beg19], and is one of the biggest CDN providers in the world. Akamai has deployed around 288 000 servers in 136 countries [Akandc].

Google operates their own backbone network [TP19], and has many data centres available through edge POPs [Goondc]. A POP is a geographic location at the edge of the network. Google is world-spanning, and is present at around 90 Internet Exchange Point (IXP) and at over 100 interconnection facilities [Goondc]. Google has cloud solutions, and owns content (e.g. YouTube). The content is efficiently delivered to the users by CDN technology. They also offer commercial CDN solutions, which smaller companies can buy to make their content more available [Goonda]. Google can be described as a combination of cloud CDN, content CDN and commercial CDN.

2.4 Uninett and the use of CDN technologies

This section will give more details about the Norwegian ISP Uninett, and how CDN technologies are used in the network.

Uninett is a Norwegian ISP funded by the government, which task is to provide the National Research and Education Network (NREN). In addition, they provide different services, like the login solution FEIDE and wireless access solutions like eduroam [Uninda]. Uninett calls themselves the digital foundation for the research and education community in Norway, and connects the Norwegian research and education network to the international research and education community [Uninda]. Uninett cooperates both with NORDUnet and GÉANT [Unindd]. GÉANT is the European network with the purpose of connecting all the European NRENs. NORDUnet is a collaboration network between the five Nordic countries Norway, Sweden, Denmark, Finland and Iceland. Uninett has several direct connections to NORDUnet, and the link with the highest load terminates in Oslo.

Figure 2.4: Uninett's logo [Unindc]

It is an important responsibility to be the digital foundation of research and education in Norway. Norway is a highly developed technological country, and was the first country outside the US that connected to ARPANET [BBDN17]. According to statistics from 2018, 96,5% of the population uses Internet [FN], and 90% uses Internet on a daily basis [Kom18].

Uninett's network has high capacity, and the link capacity is either 10 or 100 Gbps [Unindb]. The area of research and education has high requirements to a network and its capacity, e.g. due to the amount of data that is being processed, stored and transported [Unindb].

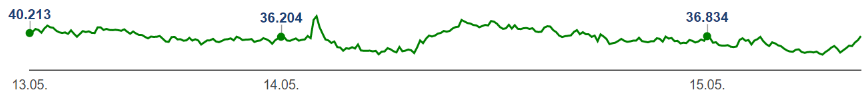


Figure 2.5: The amount of traffic in Uninett in Gbps [Unindb]

Uninett has already implemented CDN technologies in their network, to better meet the requirements. The Norwegian people are very interested in winter sport like e.g. cross country skiing and biathlon. In 2011 the FIS Nordic World Ski Championship was arranged, and the different races were broadcasted live through NRK [1]. This increased the load in Uninett's network so much, that they started to look for solutions to avoid the possible high load on bottleneck links [Eid20].

Uninett became a part of Akamai Accelerated Network Partner (AANP) program, which is a program that is meant to benefit both the ISP and CDN provider by locating servers from the CDN provider inside the ISP's network. Uninett has two Akamai edge servers in their network, one located in Oslo and one in Trondheim. These servers were installed in the network in 2011 [Eid20]. Akamai provided hardware, and controls which content the node holds, and all other configuration. Uninett connects the nodes to their network and provides space, power and cooling for the hardware.

In addition to this, in April 2020 Uninett and Microsoft established a private peering collaboration at Digiplex (a private peering facility in Oslo) [Ell20]. Other relevant peering agreements is the public peering to Amazon, Cloudflare and Dropbox through the Norwegian Inernet Exchange (NIX) in Oslo.

NORDUnet has several agreements with different CDN providers. They participate in the AANP program, and they host a Google Global Cache (GGC) in their network. GGC is similar to AANP. Google provides necessary hardware, and the ISP will provide space and power to the node [Goondb]. They also participate in Netflix's program called Netflix Open Connect [Netnda]. The ISP can either peer with Netflix through Private Network Interconnect (PNI) or install embedded Open Connect Appliances (OCA) in the ISP network. NORDUnet has peering agreements with CDN providers like Facebook, Twitch, Apple and Amazon, based on the network map available through [Uni20].

[1]NRK is a Norwegian governmental owned broadcaster

2.5 CDN from an ISP's point of view

This section will focus on the benefits of using CDN technologies from an ISP's point of view. It has been mentioned briefly earlier in this chapter, and will be described further.

One of the main goals of an ISP is to provide an optimized network [FPS+13]. At the same time the ISP can not control when traffic appears, from which source and to which destination. The ISP has to control its network in the best possible way by having a good overview of normal traffic patterns and by implementing solutions to handle expected and unexpected traffic (like flash crowds). Collaborating with CDN providers can help to handle challenges like flash crowds.

One of the benefits with collaborating with CDN providers, is that upgrading the network infrastructure may be delayed [FPS+13]. This can contribute to saving costs.

Several CDN providers like Google [Goondb], Akamai [Akanda] and Netflix [Netnda], offer different programs or peering options to ISPs. These solutions often come with a set of requirements, e.g. to the amount of traffic that is exchanged between the CDN provider and the users in the ISP network, link capacity and service center availability. Agreements about peering or participating in CDN provider programs may come without direct billing between the CDN provider and ISP. A CDN provider program may involve hosting hardware inside the ISP network, and the ISP has to cover the costs for power, cooling and connection to the network, in addition to providing space in data centres or network locations. This is associated with costs, so such an agreement can not be seen as free for the ISP. Both the ISP and the CDN provider cares about the user experience and the delivered Quality of Service (QoS), both peering and programs can contribute to this goal and the collaboration can be a win-win for both parties. Especially, since it may involve reduced costs [FPL+13].

Possible collobaration types can be uncollaborative, managed CDN, licensed CDNs and telco CDN [HNC+16]. Uncollaborative means there is no collaboration between the ISP and the CDN provider. Managed CDN is when the CDN provider puts servers inside the ISP network, and is the most relevant type for this project. When a CDN provider sells CDN services to an ISP, and the ISP can sell them to their customers, it can be called licensed CDN. Lastly, Telco CDN is when the ISP makes own CDN solutions, e.g. like Telenor CDN [Telnd]. In addition to these types, peering agreements may be an option.

There are several other solutions that at least exist in the literature, that aim to make the collaboration between CDN and ISP better. One of them is implementing

Network Function Virtualization (NFV) [HNC+16]. The components that are best to implement into the ISP network is servers storing content and/or caching it, and management entities like routers.
Content-aware Traffic Engineering (CaTE) is another approach, with the main goal of solving the server selection problem in CDN, and to exploit path diversity [PFS+12]. Both the ISP and the CDN cooperate to achieve the goal, instead of only the CDN choosing server without having information about the network traffic (which the ISP has).
Network Platform as a Service (NetPaaS) is a way of how servers can be implemented in the ISP network, and is based on CaTE [FPL+13]. Where the servers are placed and which server that serves the user, are the main parts in this solution. It is a cooperation where both the CDN provider and ISP help each other to find the best solution.

2.6 Tools

This section is based on the research done in the pre-project of this project [Asp19], and will provide the background for the choice of tool to measure the amount of CDN traffic.

Uninett uses IP Flow Information Export (IPFIX) to collect network traffic. This is a protocol that is based on the Cisco developed protocol NetFlow [Cis12]. IPFIX is developed by IETF [For13], and can be configured to collect traffic from e.g. the different interfaces on a router. IPFIX will be used in this project to collect the network traffic, and is implemented at Uninett.

Which tool that should be used to analyse the traffic demanded some research. The tools that were discussed and considered during the period of the pre-project was:

- Elastic Stack
- SiLK
- Kenitk
- Plixer's Scrutinizer

In addition to these, Humio was also considered in the beginning of this project. Based on the pre-project SiLK, Elastic Stack and Kentik was considered as likely tools to use. SiLK was already available at Uninett, and considered as a possibility, but not the main choice. Elastic Stack was also available at Uninett, and with visualization possibilities included, it was considered as the most likely tool to use. Kentik would have been the first choice, because of the implemented CDN Analytics, if it was possible to make it available at Uninett, but it was condemned unlikely.

The analysing tools used in this project is SiLK and Kentik, both Elastic Stack and Humio were unavailable during the project period. SiLK was the only analysing tool that was implemented in Uninett and available during the project period, while Kentik is implemented in NORDUnet and access was granted through the supervisors of this project. The following description of SiLK is reused from the pre-project [Asp19]:

The tool SiLK, or System for Internet-Level Knowledge, is a tool developed by CERT Network Situational Awareness Team (CERT NetSA) for analyzing network traffic [Netndc]. The tool is command line based, and is free for download. SiLK converts data from NetFlow or IPFIX to a more efficient format, so that big amounts of data can be processed and analyzed. This tool is easy to understand, since it is a command line tool. SiLK is not a good solution when it comes to visualizing the results, but can be a good tool to start with to understand the captured data.

Kentik describes themselves as ”The Network Traffic Intelligence Platform”, and uses Artificial Intelligence (AI) to provide real-time analysing of the network [Ken]. The insights are supposed to help optimizing and making the network more efficient. Kentik can be used through a web interface, and the graphics make it easy to instantly get a good overview. In contrast to SiLK, exactly how the numbers are achieved is harder to understand. Kentik has integrated a CDN engine, which has implemented info about several CDN providers, so that the traffic can be categorized as CDN traffic. This info is supposed to help with traffic engineering the CDN traffic in the network, and optimizing the network [Ken]. They have mapped 54 CDN providers, as of February 2020.

2.7 ICN

As mentioned in Section 2.1, communication have been an important area of usage in the Internet, especially in the beginning. The Internet architecture is based on this, and it is host-centric. The development towards content being the main area of usage, makes it relevant to challenge the traditional host-centric architecture [KP18]. The users do not care where they get the content, as long as they get it. In addition, the continuous expanding Internet have introduced several add-on techniques to handle the increase, CDN being a major one. This contributes to a complex network with many variations in technologies and solutions. Other issues are e.g. scalability, for example IP-address allocation and availability. IPv6 mitigates this problem, but the flexibility of the host-centric architecture is still limited [KP18].

Information Centric Networking (ICN) is an umbrella term for projects which looks into new architectures where the key word is content [KP18]. There are many projects, and terms like Content Centric Networking (CCN) and Named Data Networking (NDN) are related to ICN. New proposals may be based on in-network caching, and the purpose is to move away from the connection between user and content provider. One cruical part in content-centric architecture is the identification of the content, which needs to be ”globally unique and independent of the location where they are stored” [KP18].

The concept of ICN can be like the one illustrated in Figure 2.6. When new content is made available, this is advertised by the content provider. If users are interested in some content, they put the interest out in the network. The users are not aware where the content is, or not even who originally made the content available. When such an interest matches with an advertisement, a path for delivery will be set up. This most likely includes that the content has to pass through several middle nodes to reach the user. All the nodes in the delivery path can choose to cache the content for later interests from users. One important aspect is that the content itself must include some proof that it really is the advertised content. The trust is between the user and content, not user and publisher.

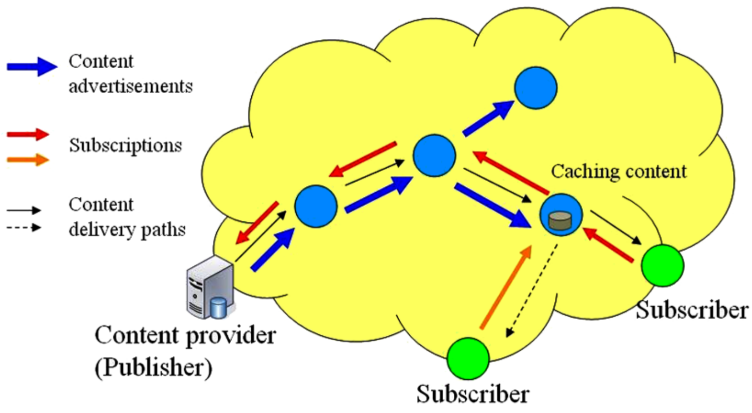


Figure 2.6: An example of how ICN architecture can look like [KP18]

Chapter 3 Methodology

This chapter describes the method used in this project. The chosen method is Design Science. The following sections are based on information available in [Wie14].

3.1 Design Science

The method used in this project is Design Science. Design Science fits well with computer science projects, and is based on investigating a problem, and designing a treatment to the problem.

There are several reasons for why Design Science is well suited as the chosen method. In Design Science real world problems are investigated, and the solution is not limited to one existing solution. In this master thesis CDN technologies are already implemented at the studied ISP, and therefore a ”solution” already exist. The goal is to improve and redesign the already implemented solution.

In Design Science some important concepts are artifact and context. An artifact is something created by humans, and the main artifact in this project is the model for how the use of CDN technologies can be optimized in Uninett. The context is Uninett’s network. It is the interaction between the artifact and the context that tells if it is a good solution or not. This means that the same artifact does not necessary give the same result in a different context. The interaction between the model and Uninett’s network should be as efficient as possible.

Design Science can be conducted through a design cycle, consisting of several steps in a cycle: problem investigation, treatment design and treatment validation. The design cycle is a part of the engineering cycle, Figure 3.1 illustrates this.

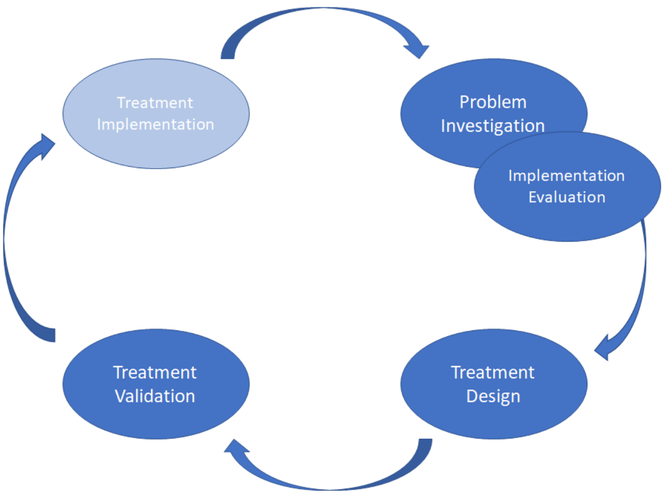


Figure 3.1: The engineering cycle

The next paragraphs describes the different steps in the design cycle in relation to this project.

3.1.1 Problem Investigation (and Treatment Evaluation)

The Problem Investigation defines a considerable amount of the work in this project. The pre-project [Asp19], conducted in the Fall semester in 2019 was the beginning of the Problem Investigation.

Further on, a literature study was performed in the beginning of this project, especially focusing on understanding CDN at a deeper level. In addition, future technologies like ICN and possible ways to use CDN technologies were looked into. The design cycle is iterating, and the literature study has been performed whenever more knowledge has been needed, even if the defined step is another than Problem Investigation.

As a part of the Problem Investigation another design cycle can be defined. This can also be defined as an observational case study. One important part of this project has been to produce results of the amount of CDN traffic in Uninett, so that the optimization could be based on actual traffic volumes. This is a quantitative method. The artifact is the method used to measure, and the context is Uninett's network. This was an iterating process, and different approaches were assessed. The resulting one was using the analysis suite SiLK, where the choice was made based on tool availability. Several scripts have been made and executed, and whenever errors were discovered, new scripts were developed. Validation of the results has been done by comparing to similar networks (NORDUnet), and by experts at Uninett. Based on the measurements of the amount of CDN traffic, an analysis has been done to build the foundation for the resulting model. Microsoft Excel has been used for this purpose.

CDN technologies are already implemented in Uninett. To be able to improve the implementation, a good understanding of the existing implementation is crucial. The engineering cycle is an expanded design cycle, where Implementation Evaluation is a step in the cycle (see Figure 3.1). Normally this step is not conducted in a Design Science project, due to that the treatment is not implemented in the real world context as a part of the project. The evaluation is normally done after a treatment is implemented. In this project it is important to understand the already implemented model, and therefore the Problem Investigation also contains the step ”Implementation Evaluation”. It is a requirement to understand the implemented model, to be able to improve it.

3.1.2 Treatment Design

The Treatment Design in a design cycle is the resulting design of the artifact. In this project, the treatment is the recommended model of how the use of CDN technologies in Uninett can be optimized. Through this step the Problem Investigation-step had to be revisited several times to gather all the information needed, to be able to design the best possible treatment. Especially, the defined criteria and the results from the measurements and analysis, contributed to the design of the model.

3.1.3 Treatment Validation

Validation of the designed treatment is a vital part of the project, to be certain that the presented model actually improves the problem, and not just changes or redefines it. The method chosen to validate the treatment is expert opinion. This means that the model is presented to experts in Uninett, and they provide an assessment based on the defined criteria. The expert group consisted of five engineers from the department operating Uninett’s backbone: Frode Storvik, Einar Lillebrygfjeld, Runar Borge, Håvard Eidnes and Svein Ove Undal. It is really important that the experts understand the artifact, so that they can predict how it will interact with the context. This can be a challenge, and demands a good presentation to relay a good understanding. The validation was done in one meeting, where the appraisal gave that it was no need for another meeting.

3.2 Method assessment

The method has been used as a tool to understand the problem, and how to solve it. By using the framework of Design Science, finding a solution to the problem has been done in an orderly manner. Splitting the problem into several parts have also been aided by the knowledge of Design Science. The iterative approach has been an advantage in the process, and has contributed to finding a good solution by learning from errors and mistakes. Design Science also fits very well with improving an already existing solution, which is what this project has attempted to achieve. A disadvantage with the method, especially regarding the observational case study, is the limited time and limited measure points. This may have impacted the measurement results, which was the foundation for the recommended model.

Chapter 4

CDN traffic in Uninett

This chapter will describe how the measurements and analysis have been done, and in the end the results will be presented. The files used to obtain the results, and the results itself, are made available in a GitHub-project [Asp20].

4.1 Measurements

SiLK is the chosen tool to obtain results of how much CDN traffic it is in Uninett. As mentioned in Chapter 2, SiLK is a command-line based suite of tools, and provides several possibilities to analyse network traffic. To get to know the different possibilities, the handbook ”Network Traffic Analysis with SiLK” [KOSS19] provides a good knowledge, in addition to directly contacting CERT NetSA [Hor20].

SiLK processes binary data, which makes it an effective tool to handle big amounts of data. Most of the analysis exploits this advantage, so that handling big amounts of data is done in a timely manner. The binary files contains SiLK Flow records, which is converted from collected IPFIX records. This convertion is done by Uninett. SiLK is installed and configured by Uninett. The data set containing flow records used in this project is collected from the router oslo-gw1, which is a central node in Uninett. Both the links to NORDUnet in Stockholm and to the Norwegian Internet Exchange (NIX) in Oslo, is terminated at oslo-gw1. Uninett conducts collection of IPFIX records from their nodes approximately at midnight each day. The collection is done one node at the time, in 2 minutes samples, which means that it is not done exactly at midnight for every node. This leads to an asynchronous sampling.

SiLK characterize traffic based on the direction [KOSS19]. The main types used in this project are in, inweb, out, outweb and ext2ext. Uninett characterize in and inweb as all traffic with destination as AS224, which is Uninett’s Autonomous System Number (ASN). Out and outweb is traffic with source AS224, and ext2ext have another ASN both as source and destination. In addition to these, int2int is used to validate the results in terms of total traffic. int2int is traffic where both source and destination ASN is Uninett.

The main approach is based on [Hor20], which is to use ASNs to filter the CDN-traffic. The measurement period is chosen to be March 10th and Match 11th 2020, which are two normal businessdays, and should provide a good overview of the traffic in Uninett. Weekends are not that relevant due to that Uninett is a NREN, and based on [Uni20] the traffic volume is higher during businessdays than weekends. In addition, March 12th was the day when the Norwegian government broadcasted the new guidelines in relation to the pandemic covid-19 [KS20]. This changed the traffic pattern, and results would have been highly impacted by the new way of life.
By the help of the tools rwfilter, rwuniq and rwcount, the traffic will be filtered and converted to a human-readable format. The following paragraphs describe how this is done.

First of all, the selection of CDN providers must be carried out. To do this, Kentik CDN Analytics is used as inspiration. Access to Kentik CDN Analytics was provided through the supervisors of this project. Kentik is used by NORDUnet, and NORDUnet provides data from their network to Kentik. One assumption that is made, is that the big CDN providers in NORDUnet most likely are big in Uninett. This is due to NORDUnet being a transit-network for the NRENs in Norway, Denmark, Sweden, Iceland and Finland, as mentioned in Section 2.4. The following CDN providers have peaks around 1 Gbps in NORDUnet[1], and each of these CDN providers will get their own measurement results in this project. It is 16 main CDN providers:

> Akamai, Amazon, Apple, Cloudflare, Dropbox, Facebook, Fastly, Google (YouTube), Level3 (CenturyLink), Limelight, Microsoft Azure, Netflix, Reflected Networks, Stackpath, Twitch.TV and Verizon's Edgecast

The following CDN providers are small in NORDUnet, and do not have high traffic peaks. In case these are bigger providers in Uninett than in NORDUnet, they are filtered as one collection, to see if any of them should be analysed as one of the main CDN providers. It is 24 smaller CDN providers:

> Alibaba Cloud, Azion, BelugaCDN, BunnyCDN, CacheFly, CD-Networks, cdnnow, CDNvideo, CDN77, ChinaCache, G-core CDN, Imperva Incapsula, Instartlogic, Internap, Kingsoft Cloud, LeaseWeb CDN, Medianova, Ngenix, Pandora, QUANTIL China

[1]March 2020

NetCenter, Tata Communications, Tencent Cloud, Yahoo and Zenedge

All the CDN providers have one or several ASNs associated with them. Kentik keeps an overview of the ASN for each mapped CDN provider, and the ASN list in this project is based on that overview. In addition, the website operated by Geoff Huston, chief scientist in Asia Pacific Network Information Centre (APNIC) [APN], has been used to cross-check the listed ASNs [Husnd]. This mapping between ASN and name, have also been used to look up the CDN providers by names to see if ASNs were missing from the list. Further, for the main CDN providers, the ASN have been looked up in PeeringDB [Peend] as another cross-check. The ASN list is available in Appendix A.

rwfilter is a tool which can filter SiLK Flow records based on a set of criteria. In this project the criteria is ASN, and this is not one of the options in rwfilter. To solve this, pmap-files may be used. In this case, a pmap-file mapping ASN to IP is used. Carnegie Mellon University have a project where they collect BGP routing data from two projects: University of Oregon Route Views project and the RIPE NCC's RIS project [Insnd]. One of the outcomes of the project is historical pmap-files, which is used in this project. There exists one pmap-file for each day, as far back as to 2009, that can be downloaded. The measurement period for this project is March 10th and 11th 2020, and the respective pmap-files are used: 20200310.bgp.ripe.pmap and 20200311.bgp.ripe.pmap.

One drawback with rwfilter and the use of pmap-files, is that the filter-request fails if the ASN do not exists in the pmap-files. Due to this, this must be checked beforehand. Since there is one pmap-file for each day, this must be done for both files. To check this, the following method can be used:

```
rwpmapcat 20200310.bgp.ripe.pmap | grep -w 'AS224'
```

rwpmapcat is a part of the SiLK tool suite, and it prints the pmap-file in text. By using "grep" all lines with the noted ASN will be printed. This is a cumbersome method, when the amount of ASNs that need to be checked is approximately 240 per file. Instead, awk is used to make the ASN-check more efficient:

```
awk -f asncheck1.awk 20200310.bgp.ripe.pmap
```

This method prints "1" whenever the ASN exists in the file. The files "asncheck1.awk" to "asncheck6.awk" is available at the GitHub-project [Asp20]. In Appendix A, all the ASNs in bold exists in the pmap-files for March 10th and March 11th 2020.

rwfilter can now be used to achieve results for each CDN provider. The measurement process flow is illustrated in Figure 4.1. The amount of SiLK Flow records is large, and UNIX pipes are used to make the filtering-process more efficient. UNIX pipes are used to create a chain of rwfilter commands, where the unrecognized output of one rwfilter command (”fail-results”) are fed into the next rwfilter command for further filtering. This is time-efficient, since there is no need to filter the whole data set for every rwfilter command and CDN provider. Several bash-scripts are used to get the results, and Table 4.1 gives an overview of the different scripts. One example script is available in Appendix B, and the rest are available through the GitHub-project [Asp20]. The following part is an example of how rwfilter is used in one of the scripts.

```
rwfilter --type=in,inweb --start=$START --end=$END --
    pmap-file=FACEBOOK:$pmapfile --pmap-src-FACEBOOK=
    AS32934,AS54115,AS63293 --pass=passinfb.rw --fail=
    stdout --data=$DATA | rwfilter --type=in,inweb --
    site-config-file=$SILKCONF --pmap-file=NETFLIX:
    $pmapfile --pmap-src-NETFLIX=AS2906,AS40027 --pass
    =passinnetflix.rw --fail=stdout --input-pipe=stdin
     |
...
...
rwfilter --type=in,inweb --site-config-file=$SILKCONF
     --pmap-file=ROKLA:$pmapfile --pmap-src-ROKLA=
    AS54994, ... ,AS40428 --pass=passinrokla.rw --fail
    =failinrokla.rw --input-pipe=stdin
```

This is from the script ”rwfilterin2020031X.sh”, and it filters out all traffic with source ASNs as noted. It is source ASN due to that the destination ASN for the types in and inweb, is AS224. There are several variables in use, all noted with a ”$” at the start, which makes the script easier to use when results from another date or time are needed. The last rwfilter (”rokla”) is a collection of all the smaller CDN providers, and the complete fail-situation is gathered in a SiLK-file. Every pass-situation is also gathered in SiLK-files, that can be used in further operations.

rwuniq and rwcount are used to get the results into human readable-format, and to exclude IP-addresses so that the user privacy is protected. More about user privacy in Section 6.4. rwuniq are used to sum up bytes per ASN belonging to the CDN provider in the defined time span. This is especially needed information to check the collection with smaller providers to see if any of them should be defined as main CDN providers. rwcount counts all the bytes in bins, where a bin represents a time interval. In this project the bin-size is 120 (2 minutes). This is chosen because

Uninett samples every 2 minutes in their statistics. The column-separator is chosen to be ";", which makes it easy to convert the .txt-files into .csv-files to be able to analyse the results with the use of Microsoft Excel. The following example shows how rwuniq and rwcount are used:

```
rwuniq passinfb.rw --pmap-file=FACEBOOK:$pmapfile --
   fields=src-FACEBOOK --values=records,bytes,packets
    --no-col --output=passinfb.txt --copy-input=
   stdout | rwcount --bin-size=$BIN --no-col --column
   -separator=";" > countinfb.txt
```

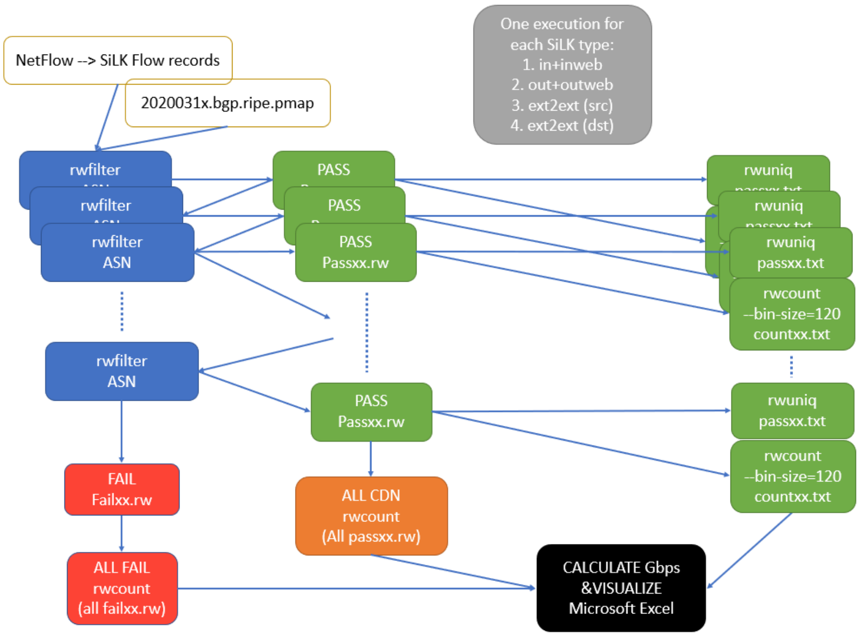

One execution for each SiLK type:
1. in+inweb
2. out+outweb
3. ext2ext (src)
4. ext2ext (dst)
NetFlow --> SiLK Flow records
2020031x.bgp.ripe.pmap
rwfilter
rwfilter
rwfilter ASN
PASS
PASS
PASS Passxx.rw
rwuniq
rwuniq
rwuniq passxx.txt
rwcount --bin-size=120 countxx.txt
rwfilter ASN
PASS Passxx.rw
rwuniq passxx.txt
rwcount --bin-size=120 countxx.txt
FAIL Failxx.rw
ALL FAIL rwcount (all failxx.rw)
ALL CDN rwcount (All passxx.rw)
CALCULATE Gbps &VISUALIZE Microsoft Excel

Figure 4.1: The measurement process flow

To be able to find out how much of the traffic in Uninett that is CDN traffic, rwcount are used to sum all the CDN traffic for each of the three categories (in/inweb, out/outweb, ext2ext), and to sum them all. This also gives an overview of the traffic distribution in 24 hours. One text-file for each category, where all the pass-filenames are listed, is used as an argument to rwcount. One text-files with all the pass-files is also made to get the total CDN traffic for each day. The script does not differ in filenames, so the results are separated by following an execution order of the scripts per day, and gather the results per day in their own directory. The following shows an example how rwcount is used to achieve these results:

```
rwcount --xargs=allpassfilesin.txt --bin-size=$BIN --
    no-col --column-separator=";" > countpassallcdnin.
    txt
```

The text-file "allpassfilesin.txt" and the script "rwcountallcdn.sh" is available in Appendix C, and the other files are available through the GitHub-project [Asp20].

The process of obtaining results is an iterative process, and the results are used to discover possible deviations. One such deviation is both Google and Netflix. The first result shows very little traffic, which seems odd compared to results in Kentik. With support from NORDUnet, IP prefixes for their implemented CDN nodes were made available. Google, Netflix and Akamai have CDN nodes in NORDUnet, and these nodes are configured with IP addresses belonging to NORDUnet:

Netflix: 109.105.98.160/31, 109.105.98.158/31
Akamai: 109.105.109.0/26
Google: 109.105.109.192/26, 109.105.98.192/27

These IP prefixes are used as input to rwfilter, together with the ASN for the CDN provider. By using both ASN and IP prefixes, the results are separated, which makes sure that the result is not counted twice. The script "rwfiltercdncache.sh" is available in Appendix D, and at the GitHub-project [Asp20]. Example code is shown below:

```
rwfilter --data=$DATA --start=$START --end=$END --
    scidr=109.105.98.160/31,109.105.98.158/31 --type=
    in,inweb --pass=passinnetflixcache.rw --fail=
    stdout | rwfilter --type=in,inweb --site-config-
    file=$SILKCONF --pmap-file=NETFLIX:$pmapfile --
    pmap-src-NETFLIX=AS2906,AS40027 --pass=
    passinnetflixny.rw    -input-pipe=stdin
rwuniq passinnetflixny.rw --pmap-file=NETFLIX:
    $pmapfile --fields=src-NETFLIX --values=records,
    bytes,packets --no-col --output=passinnetflixny.
    txt --copy-input=stdout | rwcount --bin-size=$BIN
    --no-col --column-separator=";" > countinnetflixny
    .txt
rwcount passinnetflixcache.rw --bin-size=$BIN --no-
    col --column-separator=";" > countinnetflixcache.
    txt
```

Table 4.1: Scripts overview

Step	Script	File arguments
1	rwfilterextdst2020031X.sh	
2	rwfilterextsrc2020031X.sh	
3	rwfilterin2020031X.sh	
4	rwfilterout2020031X.sh	
5	rwcountallcdn.sh	allfailfiles.txt, allpassilfes.txt, allpassfilesext.txt, allpassfilesin.txt, allpassfilesout.txt
6	rwfiltercdncache.sh	
7	rwfiltertwitch.sh	
8	invalidasn.sh	
9	invalidasntoipprefix.sh	
10	verification.sh	

X is either 0 or 1 depending if results are obtained for March 10th or 11th

The original result for Twitch was another deviation. It was discovered that the ASN was wrong, and had to be corrected. Another script was made, ”rwfiltertwitch.sh”, and executed separately. The results had to be manually added to the others. The script is available through Appendix E.

Scripts are used to make the filter-process more efficient. In addition, this was necessary in this project since the execution was done by Uninett employees. Table 4.1 gives an overview of the scripts, and which order to execute them to achieve the results. This is valid for one day and one pmap-file at the time. Because of the iterating process, step 5 do not include the results from step 6 and 7. This is corrected manually, by adding the results to Microsoft Excel to achieve valid graphs and valid total values. Step 8 and 9 tries to achieve results for some of the ASNs that exists in the pmap-file, but gives an error when used in the scripts. Step 10 is used to validate the results, more about this in Section 4.2.

All the files containing the results from rwcount are .txt-files. Microsoft Excel is chosen to analyse the data further, and to visualize the results. The .txt-files are converted to .csv-files to be used in Excel. The results are presented as date and time samples (bin) and traffic volume (bytes) for each bin. The results from SiLK are then converted to Gbps by this formula (given that the bin is 2 minutes):

$$Gbps = ((TrafficVolumeinBytes * 8)/120)/(1024 * 1024 * 1024)$$

Graphs are made for each result obtained through the scripts. All the results files from SiLK can be found through the GitHub-project [Asp20].

In addition to the measurements conducted through this project, measurements of the traffic load in Uninett for March 10th and 11th have been provided by Uninett. This is measurements they do as a routine, and they are stored in a RRD-database. This is based on the same asynchronous collection of IPFIX-data as mentioned earlier in this Section. Traffic load statistics are publicly available through an Uninett website [Uni20][2]. Uninett has provided rawdata for the period of interest, so that the graph and numbers could be recreated.

Uninett has also provided the results for the traffic load in the different geographic areas in Norway (East, West, North and Mid), and for the Akamai nodes in Trondheim and Oslo. The results from the different geographic areas are provided by using the same method as for the traffic load in Uninett, while the measurements for the Akamai nodes is based on measurements on the links ifi2-gw - Akamai and teknobyen-gw1 - Akamai [Uni20]. The sampling rate is approximately every 2 minutes, and the traffic is given in kbps. The raw-files containing the results for the Akamai-nodes can be found in the GitHub-project [Asp20].

4.2 Validation

To be able to cross-check the results, all the traffic at oslo-gw1 was counted using rwfilter together with the types in, inweb, out, outweb, ext2ext and int2int. This is implemented in the beginning of ”rwfilterout2020031X.sh”:

```
rwfilter --data=$DATA --start=$START --end=$END --
    type=in,inweb,out,outweb,int2int,ext2ext --all=
    countalltraffic.rw
rwcount countalltraffic.rw --bin-size=$BIN --no-col
    --column-separator=";" > countalltraffic.txt
```

The type int2int is not considered when CDN traffic is measured, due to the assumption that Uninett’s customers do not send CDN traffic to each other[3]. Though, results are needed to be able to look at the results from all the traffic at oslo-gw1 (as mentioned in the previous paragraph). This is implemented in ”verification.sh”:

[2]https://stats.uninett.no/stat-q/r-all?q=all&name=Customers

[3]The traffic from the Akamai nodes inside Uninett is int2int, but is measured with the use of Uninett’s rawdata for the link to the nodes

```
rwfilter --data=$DATA --start=$START --end=$END --
    type=int2int --all=countalltrafficint.rw
rwcount countalltrafficint.rw --bin-size=$BIN --no-
    col --column-separator=";" > countalltrafficint.
    txt
```

Results for all the CDN traffic, non-CDN traffic and int2int-traffic can be summed up, and compared to the results that counts all the traffic at oslo-gw1.

Figure 4.2 illustrates the add-on of validation to the measurement flow.

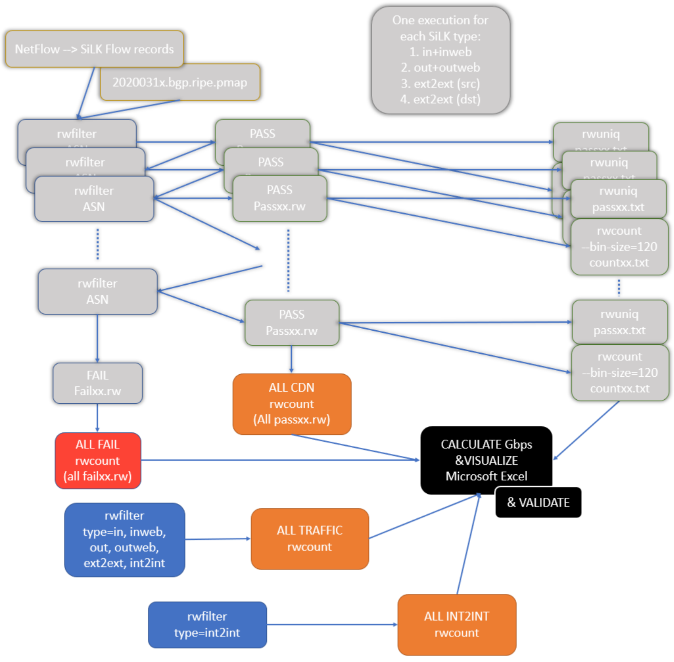

NetFlow --> SiLK Flow records
2020031x.bgp.ripe.pmap
One execution for each SiLK type: 1. in+inweb 2. out+outweb 3. ext2ext (src) 4. ext2ext (dst)
rwfilter ASN
PASS Passxx.rw
rwuniq passxx.txt
rwcount --bin-size=120 countxx.txt
FAIL Failxx.rw
ALL CDN rwcount (All passxx.rw)
ALL FAIL rwcount (all failxx.rw)
CALCULATE Gbps &VISUALIZE Microsoft Excel
& VALIDATE
rwfilter type=in, inweb, out, outweb, ext2ext, int2int
ALL TRAFFIC rwcount
rwfilter type=int2int
ALL INT2INT rwcount

Figure 4.2: Additional validation steps in the measurement flow

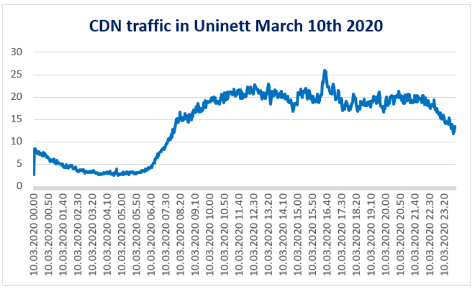
The first execution of the scripts gave errors about some ASNs that did not exist in the pmap-file. The ASN had to be removed to be able to execute the scripts. The ASNs were checked against the .pmap-file one more time. Some of the ASNs were listed together with another ASN existing in the filtering, and was disregarded. But it was three ASNs which were listed by themselves. To make sure that significant CDN traffic was not lost, some scripts were developed to make sure. This is step 8 and 9 in Table 4.1. The scripts are provided through the GitHub project [Asp20].

In addition, the script ”verification.sh” was executed to cross check that using UNIX pipes did not influence the result. This is step 10 in Table 4.1. The script is available through the GitHub project [Asp20].

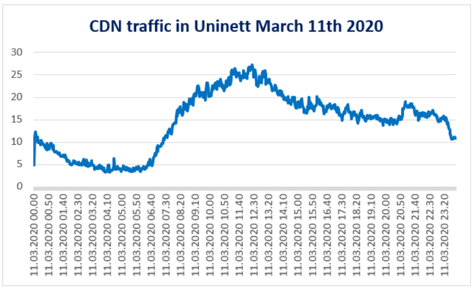
4.3 Results

The average amount of CDN traffic in Uninett mapped in this project is around 20% in total, and above 50% if only the results with destination ASN as Uninett (AS224) are considered.

The amount of CDN traffic with destination ASN as Uninett, is containing the highest load of the CDN traffic. Therefore the amount of CDN traffic in Uninett can be calculated to be based on those values. This is the traffic which uses a lot of the inter-domain link capacity in to Uninett, and is most relevant. If traffic from the implemented Akamai nodes in Uninett are excluded, the amount of CDN traffic coming in to Uninett from outside networks is 34,1%[4] (avg) and 37,7%[5] (peak). It is this amount which the model in the next chapter aims to reduce. The CDN traffic in Gbps in Uninett is visualized in Figure 4.3, and the total traffic in Uninett is presented in Figure 4.4.



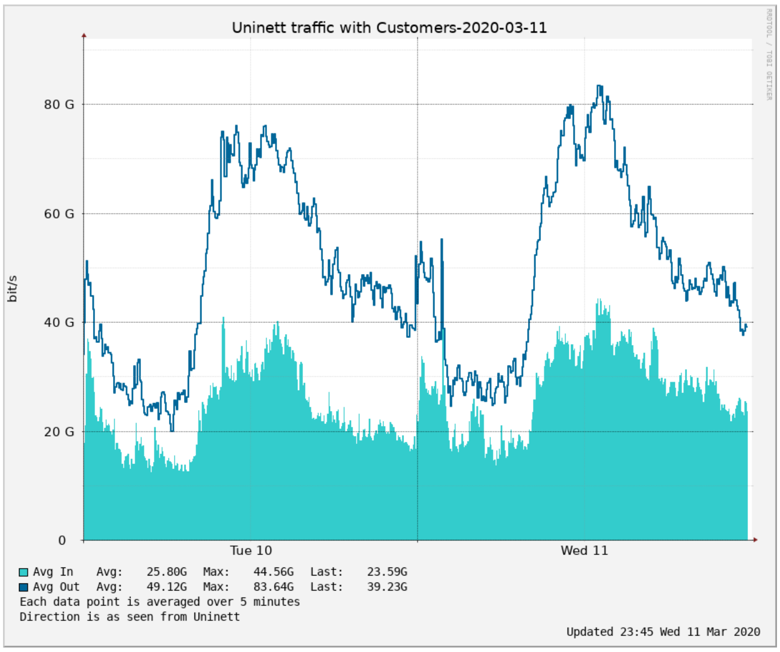
(a) March 10th



(b) March 11th

Figure 4.3: CDN traffic in Uninett in Gbps

[4] Avg CDN traffic in = 8,81 Gbps / 25,80 Gbps = 0,341
[5] Peak CDN traffic in = 16,78 Gbps / 44,56 Gbps = 0,377



```
🟦 Avg In   Avg:   25.80G  Max:   44.56G  Last:   23.59G
🟦 Avg Out  Avg:   49.12G  Max:   83.64G  Last:   39.23G
Each data point is averaged over 5 minutes
Direction is as seen from Uninett
```

Figure 4.4: Total traffic in Uninett for March 10th and 11th 2020

The average values for traffic with destination ASN as Uninett (in) is 25,80 Gbps, and 49,12 Gbps for source ASN as Uninett (out). This gives an average total of 74,92 Gbps.

The average total traffic in Uninett and the average total value of CDN traffic in Uninett, are presented in Table 4.2. The results for the type ext2ext are not presented, as it was around 0,01 Gbps and not considered significant. *Avg total* of *All traffic* is a sum of *Avg in* and *Avg out*. *Avg in, Avg out* and *Avg total* for the CDN traffic are stand-alone measurements, and therefore will summing *Avg in* and *Avg out* not give the same result as *Avg total*. The values for the CDN traffic for March 10th and 11th are average values of the results from 10th and 11th. The results for the CDN traffic consist of the SiLK filtering of the data set from oslo-gw1 and measurements provided by Uninett for the link ifi2-gw - Akamai and teknobyen-gw2 - Akamai (CDN nodes from Akamai in Oslo and Trondheim). The SiLK filtering includes the original filtering, and the results from the CDN nodes in NORDUnet (GGC, Netflix and Akamai) and from the late Twitch-filtering. There was an error in "rwcountallcdn.sh" where Twitch and Dropbox were noted twice, and therefore Dropbox had to be subtracted from the results. Twitch had a wrong ASN in the original filtering, so the result was close to zero, and did not need to be subtracted. In addition, the Akamai "in"-result at oslo-gw1 was subtracted, due to that this

Table 4.2: The average total traffic and CDN traffic in Uninett in Gbps

	Avg In	**Avg Out**	**Avg Total**
All traffic	25,80	49,12	74,92
CDN traffic March 10th	13,71	0,26	14,27
CDN traffic March 11th	13,86	0,23	14,54
CDN traffic March 10th & 11th	13,79	0,25	14,41

most likely is counted twice because that both traffic in and out from the Akamai nodes are counted. This gives the following calculation, where NORDUnet CDN nodes are GGC+Netflix+Akamai, and Uninett CDN nodes are the Akamai nodes in Oslo and Trondheim:

CDNtrafficAvgTotal = Original SiLK filtering + NORDUnet CDN nodes + Twitch-filtering + Uninett CDN nodes - Dropbox - oslo-gw1 Akamai in

For CDN traffic *Avg In*, the calculation is based on the same as for all the CDN traffic, but it uses the values for ”in”. Only the traffic coming in to Uninett from the Akamai nodes in Uninett is used, and the Akamai-traffic in to oslo-gw1 is not subtracted:

CDNtrafficAvgIn = Original SiLK filtering in + NORDUnet CDN nodes in + Twitch-filtering in + Uninett CDN nodes in - Dropbox in

The CDN traffic *Avg Out* is calculated as following:

CDNtrafficAvgOut = Original SiLK filtering out + NORDUnet CDN nodes out + Twitch-filtering out - Dropbox out

The largest part of the CDN traffic have destination ASN as Uninett, and is defined as ”traffic in”. Therefore, it can be relevant to calculate the amount of CDN traffic in. The average amount of CDN traffic in is 53,4%[6].

Kentik calculates the amount of CDN traffic based on the traffic peak-values, and to be able to compare the results, peak results are calculated. Table 4.3 shows the traffic peak values for all traffic and for the CDN traffic. The values for all traffic is based on Figure 4.4. The results for peak CDN traffic is calculated in the same way as for the avg CDN traffic based on 2 min samples.

[6]Amount of avg CDN traffic in = avg CDN traffic in March 10th and 11th / All traffic avg in = 13,79 Gbps / 25,80 Gbps = 0,534

Table 4.3: The values for traffic peaks of all traffic and traffic peaks for the CDN traffic in Uninett in Gbps

	Peak In	**Peak Out**	**Peak Total**
All traffic	44,56	83,64	128,20*
CDN traffic March 10th	23,02	1,37	26,02
CDN traffic March 11th	25,16	0,94	27,19

*The total peak is not given, but is approximately decided based on Figure 4.4, where the peaks for in and out seems to coincide

Table 4.4: The amount of CDN traffic in Uninett in percentage

	Peak	**Avg**
Total	20,7	19,2
In	53,4	51,7

Based on the values for average traffic and peak traffic, the amount of CDN traffic in percentage is presented in Table 4.4. The percentage for traffic peak is based on the highest peak registered in the period March 10th to 11th.

The traffic peak values for each CDN provider is shown in Table 4.5. The providers with traffic peaks exceeding 1 Gbps are marked with bold types. Notice that the smaller CDN providers do not give any high traffic peaks, so they have not been looked into any further. In addition, notice that ASNs belonging to BunnyCDN, G-core CDN, Tencent Cloud and Zenedge (see Appendix A), do not exist in the pmap-file, so these CDN providers have not been looked into.

The geographic distribution of traffic is of interest, to better be able to find the best placement of possible CDN nodes. This is not a very thorough approach, and it may only give a indication of where it is best to place nodes. The geographic distribution is shown in Figure 4.5.

4.3.1 Validation SiLK

The results from the attempt to validate and check the accuracy of the results provided by SiLK is shown in Table 4.6. All the values are in Gbps. Table 6.1 and 6.2 is built on these results, and gives the deviation in percentage.
The tests to check if significant CND traffic was lost because of some invalid ASNs showed that the traffic volume was insignificant or non-existing, and is not considered further.
The test to check if UNIX pipes influenced the results, showed that it did not.

Table 4.5: The traffic peaks in Gbps for the CDN providers March 10th and 11th

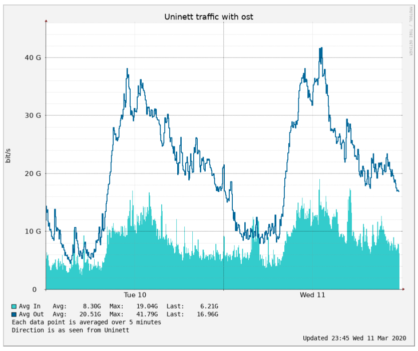
CDN Provider	March 10th			March 11th		
	in	out	ext2ext	in	out	ext2ext
Akamai*	12,66	4,51	-*	10,12	2,64	-*
Amazon	3,52	0,82	0,04	2,39	0,41	0,10
Apple	1,25	0,04	0,07	2,05	0,01	0,02
Cloudflare	1,35	0,05	0,03	0,56	0,03	0,03
Dropbox	1,28	1,07	0,02	1,21	0,59	0,02
Verizon Edgecast	1,69	0,01	0,03	1,68	0,01	0,16
Fastly	1,30	0,02	0,02	1,47	0,01	0,01
Facebook	2,80	0,12	0,01	2,40	0,08	<0,01
Google (youtube)	2,57	0,02	0,01	2,69	0,08	0,01
Level3	3,26	0,07	0,02	1,94	0,06	0,05
Limelight	0,68	<0,01	<0,01	0,72	0,02	<0,01
Microsoft	1,91	-	0,12	2,88	-	0,08
Netflix	1,97	0,03	0,06	1,77	0,06	0,02
Reflected Networks	-	<0,01	<0,01	-	<0,01	-
Stackpath	1,26	0,01	0,09	2,55	0,05	0,09
Twitch.TV	2,37	0,06	0,01	1,86	0,06	<0,01
Smaller CDN providers	0,22	0,04	0,04	0,14	0,06	<0,01

*Measurements provided from Uninett, not through SiLK, and ext2ext is not measured. ”-” denotes ”no result”.

Table 4.6: An attempt to validate and test accuracy in SiLK. Values in Gbps.

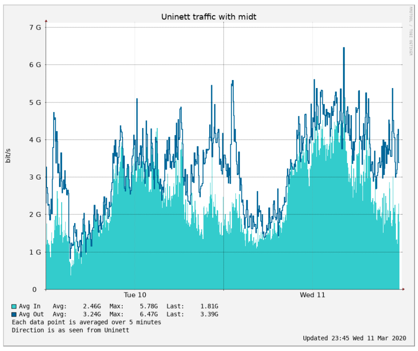
Date	**Difference peak**	**Difference average**
March 10th	2,09*	0,51
March 11th	1,41**	0,26

**(CDN + not CDN + int2int) - All traffic = 46,26 Gbps - 44,17 Gbps = 2,09 Gbps*
***(CDN + not CDN + int2int) - All traffic = 18,53 Gbps - 17,12 Gbps = 1,41 Gbps*



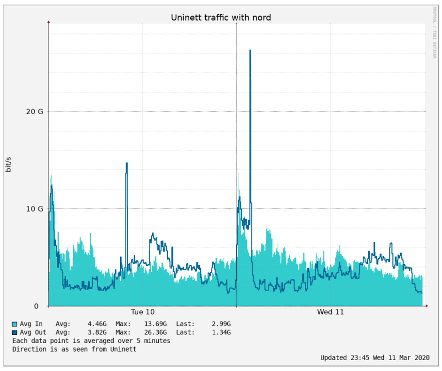
```
■ Avg In    Avg:    8.30G  Max:   19.04G  Last:    6.21G
■ Avg Out   Avg:   20.51G  Max:   41.79G  Last:   16.96G
Each data point is averaged over 5 minutes
Direction is as seen from Uninett
                                        Updated 23:45 Wed 11 Mar 2020
```

(a) East



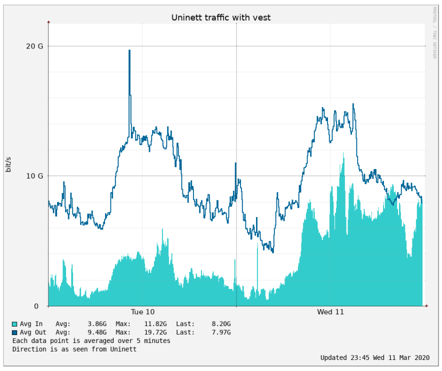
```
■ Avg In    Avg:    2.46G  Max:    5.78G  Last:    1.81G
■ Avg Out   Avg:    3.24G  Max:    6.47G  Last:    3.39G
Each data point is averaged over 5 minutes
Direction is as seen from Uninett
                                        Updated 23:45 Wed 11 Mar 2020
```

(b) Mid



```
■ Avg In    Avg:    4.46G  Max:   13.69G  Last:    2.99G
■ Avg Out   Avg:    3.82G  Max:   26.36G  Last:    1.34G
Each data point is averaged over 5 minutes
Direction is as seen from Uninett
                                        Updated 23:45 Wed 11 Mar 2020
```

(c) North



```
■ Avg In    Avg:    3.86G  Max:   11.82G  Last:    8.20G
■ Avg Out   Avg:    9.48G  Max:   19.72G  Last:    7.97G
Each data point is averaged over 5 minutes
Direction is as seen from Uninett
                                        Updated 23:45 Wed 11 Mar 2020
```

(d) West

Figure 4.5: The geographic distribution in Norway of the traffic in Uninett

Chapter 5 Model

This chapter will present the recommended model how the use of CDN technologies can be optimized based on the usecase Uninett. To prepare for the recommended model, criteria must be set. The considered solutions and the different relevant CDN providers will be presented. Lastly the model will be presented.

5.1 Criteria

The model needs to be based on a set of criteria, which highlights the improvements the model should contribute to. The criteria used, is based on dialogue with the supervisors of this project and the validation-meeting with the expert group. The following paragraphs present the model criteria:

(1) The main criteria is that the model should aim to decrease the load on central links in Uninett. This may contribute to postpone upgrading links. To be able to meet this criteria, central links are defined in Section 5.1.1.

(2) The reason why Uninett have CDN nodes from Akamai implemented in their network, is due to very high interest in watching events like FIS Nordic World Ski Championships through NRK. This can be referred to as a flash crowd situation, and leads to abnormal traffic behaviour. The model should scale to flash crowd situations. CDN providers who deliver content that may turn into flash crowd situations, are of extra interest.

(3) The recommended model should consider the economical costs it inflicts on the ISP, and is presented in Section 5.1.2. The costs should be assessed together with the advantages the model brings.

(4) The recommended model aims to be ”future proof”, which means it should endure the predicted increase in traffic volume.

(5) The model should be ”easy to implement”, and the solutions should be relevant and not over-ambitious. This also means that the model aims to be possible to implement in short-term.

(6) NORDUnet is the collaboration network between the Nordic NRENs, and it exists e.g. to gather high traffic volume to be able to establish collaborations that each of the Nordic NRENs may be too small to do. The model should consider this, and be aware that worst case scenario; big shifts in traffic may lead to disturbing NORDUnet's collaborations, which may degrade their service.

One criteria that often is considered in research and together with network optimization, is improving performance. This is not looked into in this project, and will not be considered when creating the model.

5.1.1 Central links

The central links in this project are defined to be the links that carry large amounts of CDN traffic. To define these links, the knowledge that most of the CDN traffic (around 95%[1]) have destination AS224, is used. This means that inter-domain links, or links to IXPs are of interest. The network traffic that is collected in this project is from one router, oslo-gw1, in Uninett. This router terminates links both to NIX and NORDUnet (SE). These links carry most of the traffic in to Uninett (destination AS224), seen from oslo-gw1. This means that the link oslo-gw1 - NIX and oslo-gw1 - NORDUnet (SE) carry most of the CDN traffic. In addition, the links oslo-gw1 - tullin-gw1 and tullin-gw1 - Digiplex are looked at. Digiplex is a private peering facility, and therefore these two links are central. The following stats are fetched from [Uni20] by using the map showing the traffic load, and by going into each of the central links for March 2020, the usage is roughly estimated.

– Oslo-gw1 - NIX: up to 40% usage of the 20 Gbps link capacity

– Oslo-gw1 - NORDUnet (SE): up to 40% usage of the 100 Gbps link capacity

– Tullin-gw1 - Digiplex: up to 10% usage of the 100 Gbps link*

– Oslo-gw1 - tullin-gw1: up to 25% usage of the 100 Gbps link capcity

*New link. Stats are from May 2020.

There exists other links that most likely carry some CDN traffic, like e.g. tulinn-gw1 - NORDUnet (DK). This is not looked into any further, since no measurements

[1]CDN Avg in / CDN Avg Total = 13,79 Gbps / 14,41 Gbps = 95,7%

of CDN traffic have been carried out on tullin-gw1. In addition, the assumption is that the most significant CDN providers are connected to NORDUnet in a way that the link oslo-gw1 - NORDUnet (SE) carries most of the CDN traffic to Uninett.

5.1.2 Costs

The economical costs need to be considered. There will not be given a definite answer, but this Section will attempt to highlight some costs in relation to the model. Economical costs connected to peering and in-network CDN nodes will be discussed in the following paragraphs.

Content peering can be seen as a ”free service”, when compared to transit costs where the ISP pay another ISP a per traffic volume fee for sending traffic through their network. But peering is not free, and content peering may be much more beneficial for the CDN provider than the ISP [KWS12]. Public and private peering have different costs, even though it is Settlement-Free Interconnection (SFI). The IXPs have different fees, e.g. for new connections and annual payment for existing connections. Price examples from NIX are connection fees at 10 000 NOK, while the annual fee varies depending on the capacity. For example will the first 10G port cost 30 000 NOK, an additional 10G costs 20 000 NOK and 100G costs 200 000 NOK (ex. VAT) [Excnd]. With public peering it is possible to peer with other providers present at the IXP, and it will not increase the fees (unless the capacity has to be upgraded). If the peering link is filled to a certain point (it still need to be dimensioned for peaks etc.), and several peerings have been done, then the outcome may be high. In addition, to the IXP fees the link from the ISP to the IXP may cost to establish or maintain. Peering may not always be economical gainful, and a peering break even point can be set to see when to peer and when to transit [Nor10]. Private peering may have the same type of costs as public peering. The effort needed to set up private peering is considered to be higher than for public peering, and the traffic volume needs to be high for it to be cost-efficient [Jan12].

If peering is economical beneficial for the ISP depends on e.g. the alternative transit cost. In Uninett’s case, they are connected to NORDUnet which works as a transit network. Uninett do not pay based on traffic volume, but based on Norway’s Gross Domestic Product (GDP). Not using peering may lead to the need for upgrading link capacity, so peering costs should also be considered compared to the upgrading costs. This cost may vary in size, depending on which capacity and equipment that is implemented from before. The cost of establishing new links is not considered, due to this not being considered as an option.

Uninett already has links terminated at both NIX and Digiplex, which means that this cost can be disregarded. They also have a 20G port at NIX. The recommended model can propose to utilize this port by suggesting peering-partners, without inflicting costs on Uninett. At Digiplex, each new private peering partner will demand additional costs for using a router port and purchasing access links from Digiplex. At the same time, the link in to Digiplex (from tullinn-gw1) is not utilized very efficiently. Around 10% is used at maximum [2], and it can be seen as cost-efficient to utilize this better.

The cost for hosting a CDN node inside the ISP network consists of providing rack space, power, cooling and access to the network. This may not be directly economical beneficial for the ISP, and it depends on e.g. the traffic load and the alternative transport. If not implementing a CDN node means that link capacity must be upgraded, it can be an argument for why CDN nodes are beneficial. But if the network capacity is good enough, and the transit and peering costs do not change if a CDN node is implemented, implementing a CDN node may not be an economical win situation.

This project presents a lightweight economical model which consists of two conditions. The model is inspired by [HNC+16]. The first condition (5.1) applies to the economical aspect of CDN nodes, and the second (5.2) applies to peering.

$$P < b * t \tag{5.1}$$

$$c + a < b * t \tag{5.2}$$

The following nomenclatures are used:

t	Annual transit costs
c	One-time connection fee for peering
a	Annual peering costs
P	Annual cost for a CDN node (power, space, cooling and router-port)
b	Traffic reduction factor

These conditions can be used to aid the decision if CDN nodes should be implemented, or peering established. The main purpose is that introducing new solutions should not increase the costs when compared to the costs for transit through inter-domain links. For Uninett the transit costs are static, which implies that moving traffic away from the links to NORDUnet will not reduce costs. The goal is that introducing CDN solutions should have approximately the same costs. This makes sense if the goal is to postpone upgrading links and handling abnormal traffic situations. This implies that the economical costs for upgrading links are high in relation

[2]Based on http://stats.uninett.no/stat-q/plot-all/digiplex-tullin,2020-06,day,traffic-kbit

to peering and CDN nodes. In addition, this recommendation assumes that the traffic load would keep increasing in the future.

5.2 Peering and CDN nodes as considered solutions

The different possibilities that is considered to include in the model is peering (public and private) and implementation of in-network CDN nodes (may also be called Managed CDN). Peering will only be considered if there exists possibilities in Norway. Peering ouside of Norway is NORDUnet's responsibility. These solutions are already mentioned in Section 2.5. Especially criteria 4, "easy to implement", has influenced the choice of which solutions to consider.

5.2.1 Public peering

Public peering is when different providers connect to the same Internet Exchange Point (IXP). They can choose to set up routing with other providers which are connected to the same IXP [Sch15]. NIX in Oslo, which Uninett is connected to through oslo-gw1, is an example. The advantage with public peering may be that the traffic between the providers do not have to travel through other networks to be exchanged between the source and destinationn. This may save both cost an latency. It may also be an efficient use of router ports [Jan12].

5.2.2 Private peering

Private peering is when two providers set up a direct connection between them, e.g. between their network equiment [Sch15]. This can be done in private peering facilities. One such facility is Digiplex in Oslo. Private peering fits well for bigger amounts of traffic, and when guaranteed capacity is needed [Jan12].

5.2.3 CDN nodes

CDN nodes may be a mutual beneficial arrangement between an ISP and a CDN provider. Hardware is provided by the CDN provider, while rack space, cooling, network connection and power are provided by the ISP. The CDN provider manage the content on the node. Examples like Google Global Cache and Netflix Open Connect (OCA) have been mentioned in Section 2.5. It is often content CDN provider which offer such nodes, with the commercial CDN provider Akamai as a known exception with their AANP program.

5.3 CDN providers and peak traffic

In Section 4.3, the highest traffic peaks of the CDN providers were presented. The providers with peaks over 1 Gbps, either March 10th or 11th, are presented in Table 5.1. This limit of 1 Gbps is based on the findings that several of the CDN providers requires at least 1 Gbps to approve collaboration.

These 14 providers all have peaks over 1 Gbps, and are considered as the most relevant providers to collaborate with. The following paragraphs describes possibilities for each of the relevant providers, and it is based on [Peend] if nothing else is stated. Some background, together with the CDN type and where they are present for peering is also provided. If they are seen as a provider where flash crowd situations are likely, this will be pointed out. A summary is presented in Table 5.2.

Table 5.1: Traffic peak values for CDN providers in Gbps

CDN provider	Traffic peak
Akamai	12,66
Amazon	3,52
Level3	3,26
Microsoft	2,88
Facebook	2,80
Google	2,69
Stackpath	2,55
Twitch	2,37
Apple	2,05
Netflix	1,97
Verizon Edgecast	1,69
Fastly	1,47
Cloudflare	1,35
Dropbox	1,27

Akamai. Akamai has been mentioned several times in this project, and with good reason. Akamai is the biggest CDN provider in Uninett, and one of the biggest global CDN providers [BPV08]. Most of the traffic in Uninett origins from the CDN nodes that are placed in Oslo and Trondheim, but even if this traffic is excluded, Akamai is still one of the 14. Then the traffic peak is 2,67 Gbps, and rank them as the 6th biggest. One of the services that Akamai holds, which is part of why they are the biggest in Uninett, is NRK. Flash crowd situations are likely to happen because of that. Akamai is a commercial CDN provider, and they have an open peering policy. The minimum traffic requirements are not publicly available. Akamai is present in Oslo and in Stockholm at several IXPs for both public and private peering. They also have the Akamai Accelerated Network Partner (AANP) program [Akanda], which locates CDN nodes inside the ISP network, and Uninett already got two nodes installed.

Amazon. Amazon delivers a range of different services, and can be defined as a commercial, content and cloud CDN provider. They offer the CDN service Amazon

Cloudfront, they have their own content like e.g. Prime Video and they have Amazon Web Service which e.g. offers cloud services. Amazon CloudFront has 216 POPs in 42 different countries [Amand], and is considered as a key player in the CDN market [Mar19]. Amazon has a selective peering policy, and minimum traffic peaks are not publicly available. Amazon is present in Oslo, both at NIX and at Digiplex, and CloudFront has Oslo as one of the edge locations. Uninett already has a public peering agreement via NIX.

Level3. Level3 is a part of CenturyLink, after being bought in 2017. Level3 started their CDN services in 2007 [Mar19]. CenturyLink has a global network infrastructure, and have CDN services in over 60 countries [Mar19]. They can be defined both as a commercial and telco CDN provider. Since Level3 is a part of CenturyLink, and CenturyLink is such a big global player, they have a restrictive peering policy. In addition, they require presence at multiple locations. The minimum traffic requirements are not publicly available, but it is assumed that the requirements are higher than the traffic peaks detected in this project.

Microsoft. Microsoft is characterized as a commercial, cloud and content CDN provider. Microsot Azure provide their own content like software updates, they have the CDN service Micrsoft Azure CDN, and cloud services. Microsoft Azure CDN has 130 POPs around the globe, and Oslo is one of them [Mic20]. They are present for public peering at NIX, and private peering at Digiplex. Microsoft has a selective peering policy, and minimum traffic requirements are peaks between 200 Mbps and 2 Gbps for public peering, and over 2 Gbps for private peering [Mic19]. Microsoft has a program called Microsoft Edge Caching Program for ISPs, to make the content delivery more efficient. To participate in this program, the expected daily average of downloads is around 5 Gbps [Micnd].

Facebook. Facebook owns social media platforms like Facebook, Instagram and Whatsapp, and is a content CDN provider. Facebook has over 3100 FNA nodes around the world [Bha19]. ISPs can apply to install FNA nodes in their network. Facebook decides if an ISP qualify for FNA based on e.g. if it improves the RTT, together with the traffic volume [Str20]. Facebook has a selective peering policy, and requires at least 50 Mbps peaks. They are present for both public and private peering at several IXPs in Stockholm.

Google (YouTube). Google has been mentioned earlier in this project, and is considered as a commercial, cloud and content CDN provider. Google operates a range of different services, e.g. Google cloud, Cloud CDN and YouTube. They also have their own network infrastructure. Google collaborates with ISPs e.g. by placing their GGC inside the ISP's network. Traffic requirements for hosting a GGC have not been found. Google may be a candidate when it comes to flash crowd situations due to the YouTube service. The peering policy is open, and they do not have a minimum traffic requirement for public peering. Private peering requires above 1 Gbps [Goondd]. Google is present at over 90 IXPs [Goondc], e.g. in Stockholm, both for private and public peering.

Stackpath (MaxCDN, Highwinds). Stackpath is a commercial CDN provider. They were founded in 2015, and companies like MaxCDN and Highwinds are a part of Stackpath. Stackpath requires more then 1 Gbps of traffic for public peering, and more than 5 Gbps for private peering. Their peering policy is selective [Standb]. They are present at 45 edge locations [Standa], e.g. in Stockholm for both public and private peering.

Twitch.TV. Twitch.TV is a platform for live streaming of content. For example can online games be streamed with audio of the one playing, and it is possible to chat and share the experience. Twitch is a content CDN provider. They have a selective peering policy, and requirements are not publicly available. They are present at Digiplex in Oslo for private peering, and in Stockholm both for public and private peering.

Apple. Apple provides their own content through their CDN, and is a content CDN provider. Apple has a generally open peering policy, and no traffic requirements are found. They have a solution called Apple Edge Cache, which is a node that can be placed inside an ISP network. The minimum traffic requirement is 25 Gpbs [Appnd]. Apple is present for both public and private peering in Stockholm.

Netflix. Netflix also provides their own content through CDN, and is a content CDN provider. They are one of the most popular video streaming services in the world. Netflix has a program called OpenConnect, which e.g. offers to put CDN nodes inside an ISP network. This is called embedded Open Connect Appliances (OCA), and the minimum traffic requirement is 5 Gbps peaks [Netndb]. Peering is also a part of their Open Connect, which lets ISPs connect to data centres containing Netflix OCAs. There are over 60 of these data centres around the world [Netnda]. Minimimum traffic requirements are not stated. Netflix is present for both public and private peering in Stockholm.

Verizon Edgecast. Verizon Edgecast is a commercial, content and telco CDN provider. Verizon has over 5000 network interconnections [Verndb], and they have customers like BBC, Aljazeera, CNN and HBO [Vernda]. They have an open peering policy, and are present for both public and private peering in Stockholm. Minimum requirements are not stated.

Fastly Inc. Fastly is a commercial CDN provider. They have been in the market since 2011. They have a selective peering policy, but generally open, and minimum traffic requirements are not stated. Fastly is present for private peering at Digiplex in Oslo, and for public and private peering in Stockholm.

Cloudflare. Cloudflare is a commercial CDN provider, and has an open peering policy. No traffic requirement is set for public peering, but the requirements for private peering are minimum traffic peaks at 1 Gbps [Clo]. Cloudflare is present in over 200 cities and 95 countries [Clond], e.g. at both NIX and Digiplex in Oslo for public and private peering. They are also present in Stockholm.

Dropbox. Dropbox aims to build a smart workspace [Drond], and they e.g. provide data storage to their customers. They were founded in 2007, and are a content CDN provider. They have an open peering policy, but requires minimum traffic at 50 Mbps [Dro]. Dropbox is present at NIX in Oslo for public peering, and in Stockholm for both public and private peering.

Other requirements set by CDN providers are often that the ISP has a 24/7/365 network operation centre, and that it has strategies for handling incidents. Some CDN providers require or prefer that the ISP peers at several locations, or that multiple connections are set up at the peering facility. A precondition for collaborating with the CDN provider is to keep the listing in PeeringDB up to date at all times.

5.4 Recommended model and solution

The recommended solution for Uninett is based on a model where only CDN providers with traffic peaks over 1 Gbps are considered as relevant to collaborate with. Further considerations depends on possible solutions and requirements associated with the CDN provider (see Table 5.2), together with potential traffic savings on central links. If a CDN provider is associated with flash crowd situations, it will weigh more than e.g. the economical aspect (up to a certain point). A lightweight economical model with two conditions is presented in Section 5.1.2, and is applied to some degree due to lacking information about actual costs.

Table 5.2: Overview of relevant CDN providers

CDN provider	Policy	Public peering	Private peering	CDN node
Akamai	Open	Oslo, Stockholm	Stockholm	AANP
Amazon	Selective	Oslo, Stockholm	Oslo, Stockholm	-
Level3	Restrictive	-	-	-
Microsoft	Selective	Oslo, Stockholm >200 Mpbs	Oslo, Stockholm >2 Gbps	Microsoft Edge Caching program >5 Gbps
Facebook	Selective	Stockholm	Stockholm >50 Mbps	FNA
Google	Open	Stockholm	Stockholm > Gbps	GGC
Stackpath	Selective	Stockholm >1 Gbps	Stockholm >5 Gbps	-
Twitch	Selective	Stockholm	Oslo, Stockholm	-
Apple	Selective	Stockholm	Stockholm	Apple Edge Cache >25 Gbps
Netflix	Open	Stockholm	Stockholm	Open Connect >5 Gbps
Verizon Edgecast	Open	Stockholm	Stockholm	-
Fastly	Selective	Stockholm	Oslo, Stockholm	-
Cloudflare	-	Oslo, Stockholm	Oslo, Stockholm >1 Gbps	-
Dropbox	Open	Oslo, Stockholm	Stockholm >50 Mbps	-

The following recommendation is based on Uninett's network and measurements of traffic mainly from March 10th and 11th, but the findings may be adapted and used by other similar ISPs. Before the recommendation is presented, it must be pointed out that the capacity in Uninett is good as of today. The recommendation will be heavily based on saving or moving traffic to postpone upgrading of the central links, even if the links are not over-booked as of now. The recommendation will be presented first, and the reasoning will be stated afterwards.

1. Akamai Accelerated Network Partner (AANP) Program
2. Google Global Cache (GGC)
3. Private peering Amazon
4. Private peering Microsoft
5. Public peering Cloudflare
6. Public peering Dropbox

The main criteria is to postpone upgrading links, and to be able to handle abnormal situations when the traffic changes behaviour (like flash crowds). Secondly, economical cost are considered. In addition, recommendations must handle increased traffic volume in the future, be relatively easy to implement and not cause degraded service from NORDUnet. Based on this reminder, each recommendation will be reasoned.

(1) Akamai is the biggest CDN provider in Uninett, and peaks from the already implemented CDN nodes to Uninett is at 12,66 Gbps. This amount of traffic would impact the load on the link to NORDUnet by over 10%[3], and would contribute to a more pressing need to upgrade the link. Each of the servers have 20 Gbps capacity, and it can handle abnormal situations and increased future traffic. The current costs are rack space, power, cooling and network connection, and as long as these are equivalent to the transit costs and lower than the possible costs for upgrading, it may be economical justifiable. It may be discussed if both servers are needed, but having a look at the Akamai Network Portal[4] [Akandb] shows a peak in April 2020 at 26,50 Gbps, which could not have been carried by one server. This is not inside the measurements period for this project, but it is available data, and is relevant. The recommendation is to still keep these CDN nodes in the network.

[3]Link capacity is 100 Gbps
[4]Access is provided through Uninett

(2) The traffic from Google peaks at 2,69 Gbps. This traffic comes from the GGC in NORDUnet, and through the link oslo-gw1 - NORDUnet (SE). GGC claims that 60 to 80% of the traffic may be offloaded by GGC [Goo]. Based on the link capacity and usage in March 2020, moving this traffic into a CDN node in Uninett may save over 5%[5] (given link traffic peaks at 40 Gbps). This may postpone upgrading the link. The biggest benefit may be that abnormal situations may be easily handled, and that the future increase in traffic is well cared for. It may be discussed if this is economical beneficial, and if this degrades NORDUnet's service. The economical aspect with comparing the upgrading costs, and operating expenses should be looked closer at. Given information from Kentik where the traffic peaks from Google are over 20 Gbps[6] [Kennd], this will probably not influence NORDUnet's agreement with Google. If rack space are available, e.g. together with the Akamai node, this solution is relatively easy to implement.

(3) Private peering with Amazon may move the traffic away from the links oslo-gw1 - NIX and oslo-gw1 - NORDUnet(SE). Uninett already have a public peering with Amazon, and it may be assumed that most of the traffic from Amazon travels through the NIX-link. By moving this traffic to Digiplex instead, up to 44%[7] (NIX, traffic peaks at 8 Gbps) can be freed to other public peering. This assumes that future traffic volume will increase and more peering partners appearing at NIX. The link to Digiplex is not highly used, and it can endure a 10G peering to Amazon, which should be sufficient based on the current traffic volume. Also here, the economical costs should be looked closer into, since this recommendation includes establishing a new private peering. But when looking at the possible traffic savings, this could postpone the upgrading of the NIX-link (to 100G), which also will save annual costs around 150 000 NOK (given that today's cost is the same as for 10G+10G = 50 000 NOK). This may be higher costs than a 10G private peering. Uninett already has a public peering with Amazon, and assuming most of the traffic to Uninett comes from this link, establishing private peering may most likely not impact NORDUnet's ability to negotiate good agreements.

(4) Private peering with Microsoft. This peering agreement may be especially beneficial when there are big software updates or similar, since private peering provides guaranteed capacity. Based on the peak values measured, the traffic savings can be around 7%[8] (NORDUnet (SE), traffic peaks at 40 Gbps) or 36%[9] (NIX, traffic peaks at 8 Gbps) in assumed normal situation. This may contribute to postponing upgrading of the links. The same economical argument for peering with Amazon is valid here, with saving annual cost by not upgrading the link at NIX. This also

[5](0,8 * 2,69 Gbps) / 40 Gbps = 0,051
[6]Numbers from "Last week" accessed June 8th, 2020
[7]3,52 Gbps / 8 Gbps = 0,44
[8]2,88 Gbps / 40 Gbps = 0,072
[9]2,88 Gbps / 8 Gbps = 0,36

makes the network more robust to handle increased amounts of traffic in to Uninett, by freeing capacity for both links to NIX and NORDUnet. Private peering with Microsoft has during the project period been established [Ell20]. This happened after the measurement period.

(5 & 6) Public peering with Cloudflare and Dropbox. These peering agreements already exists, and is recommended to be continued. By having public peering agreements, the traffic is likely to be routed through the NIX connection, and not through NORDUnet (SE). Given that Amazon is moved from NIX to Digiplex, the link capacity have no problem to handle these two public peerings, and the link capacity to NORDUnet (SE) can be used for other providers who are not present in Oslo. A lot of the CDN providers are present in Stockholm, and public peerings in Oslo ”frees” even more capacity on the NORDUnet link to handle the traffic load for other peering agreements (like Apple, Twitch etc.). This recommendation is based on an assumption that the traffic will increase in the future, and more peering partners will appear at NIX. Then it may be economical beneficial to maintain the presence at NIX.

As seen in Figure 4.5, most of the traffic load in Uninett is located in the area East in Norway. These results were planned to provide a foundation for deciding where CDN nodes should be implemented. It would be easy to assume that CDN nodes should be placed where the traffic volume is highest. Numbers from Akamai Network Portal [Akandb] from February 2020[10] that provide information of the use of each of the CDN nodes in Uninett, show insignificant variations in usage between the two nodes. The results may slightly favor the node in Trondheim as the most important, which does not reflect the assumption. Unplanned placement may work very well, and sophisticated methods are not necessary beneficial [SVS13]. Performance tests can be used to decide where CDN nodes is best placed.

5.5 Alternative solutions

One of the criteria was that the recommended model should be realistic to implement short-term. This Section will provide a brief introduction to some other alternatives with a more future and long-term approach. One criteria for this section is that the proposed alternative solutions should work together with the ISP’s traditional IP network.

As presented in Section 2.7, ICN is a new approach to the Internet architecture, and there exists different methods to use a content centric architecture. It is possible to add the ICN-part to the already existing host centric approach. One proposed solution is to use Smart Control Plane (SCP) [STUA18], which can be used without

[10]Chosen due to the result for March 2020 is impacted by Covid-19

the ISP having to make big changes to the IP network. The main purpose of SCP, and which distinguish it from Software Defined Networking (SDN), is the mapping between IP addresses and content naming, and the caching allocation. Both which are important areas in ICN. Introducing a SCP into the ISP network can be a possibility, and can improve the content delivery.

MicroCDN is another interesting approach [IMR14], based on using in-network caching in routers, both in the ISP core network and access network. Very little storage capacity is needed to give effect. By using 100MB memory in routers in the access network, the traffic load can be lowered with 25% [IMR14]. A challenge is encrypted web-traffic, but this can be mitigated by using a CCN approach.

It is also possible for the ISP to have their own CDN solutions. This can be implemented like Licensed CDN, as mentioned in Section 2.3 and 2.5. The ISP buys CDN solutions from a CDN provider, but operate the solutions themselves [PSR14].

Other solutions may be implementing caches in the ISP network. MicroCDN has already been introduced, but there exist many proposed solutions how in-network caching can be implemented.
A lightweight approach is presented in [TCLP13], and suggest to implement caches at the edge in the network. By taking control over the CDN part, the ISP can decide where content should be placed, which content that should be cached and which of the caches that should serve the users. This, together with knowledge of the network, makes the ISP have all the information to make the content delivery as efficient as possible. Which also can contribute to lower load on inter-domain links, which may postpone upgrading the links and can be economical beneficial [TCLP13].
The research of this lightweight approach does not consider content size, and assumes a fixed size. The content size may be big, and then the lightweight approach and MicroCDN may be inadequate. An ISP can deploy few servers with high capacity in the core, with the purpose of serving the bigger content, referred to as I-CDN [KMK$^+$13]. This approach may make the content location easier, due to the few servers where content is located.

Of these future and alternative solutions, implementing in-network caching seems closest to realistic for Uninett. Especially, the lightweight approach may be realistic without having to replace hardware (depending on the specifications of the routers in use). In a more future scenario, SCP seems like a good solution for Uninett to implement in the prospective transfer between host-centric and content-centric architecture.

Chapter 6 Discussion

This chapter will discuss the findings in this project, both based on the measurement process, the results and the presented recommended model.

6.1 Measurements

First of all, timings must be mentioned in this context. The initial plan was to collect data from a week in February, and then use several weeks to analyse the data until a satisfying result was achieved. Because of circumstances inside Uninett and technical issues regarding access to data and tools, this was not conducted as planned and caused a slow start. Several adaptions have been made, like choice of tool, when and where to measure etc. This have probably impacted the results in several ways, which will be discussed further.

The first adaption was the choice and use of the tool SiLK. SiLK is a simple, but powerful tool. The choice was made based on this being the only available tool, even though Elastic Stack was seen as the preferred choice. This may have limited the options of how results could be achieved. The advantages with this tool is that it is very ”simple” and the results are not hidden behind graphic visualization and built-in functionality. It can handle big amounts of data in a reasonable amount of time. The execution of the scripts has been done by employees in Uninett (to protect the user privacy), so time efficiency has been important. The use of UNIX pipes combined with SiLK has proven to be a good choice. Disadvantages with SiLK, may be that handling the results and presenting them, is not done by SiLK. In addition, the format is standard, and when something is adjusted, like e.g. counting the amount of bytes based on ASN, this result can not be reused by SiLK. SiLK does not store the ASNs collected through IPFIX in the SiLK Flow records, which created extra work by having to use pmap-files.

The filtering of the traffic to characterize CDN traffic, was mainly based on ASN (exceptions are the Akamai nodes in Uninett and the caches in NORDUnet). This method was chosen because it was easy and time efficient to implement in SiLK. This method does not secure that all the traffic is CDN traffic, but is based on the assumption that most of the traffic from the providers who own the ASN, is CDN traffic. Alternative ways to characterize CDN traffic could have been to base the filtering on IP prefixes, but to make this approach different from the ASN approach, it would have required making an overview of all IP prefixes associated with only CDN traffic. This list may be difficult to produce, due to that it most likely would have required cooperation from every CDN provider. Another alternative considered in the beginning of the project, was to use recursive DNS, and to create a list of keywords associated with CDN providers, to filter out the traffic. SiLK has a tool called rwresolve, which performs reverse DNS look-ups from the IP addresses in the SiLK Flow records. The results could then be checked against the list with keywords. This approach was disregarded because of not being efficient. With the amount of records, e.g. 8 billion records March 11th, the time available did not allow this approach. A test using 100 000 records was interrupted after 6 hours without finishing. Another shortcoming with using rwresolve is that it only lists the first host name that matches, which is a drawback if there are several host names belonging to an IP address [KOSS19].

Another limitation with using ASN, is that CDN nodes inside ISP network is not discovered. These CDN nodes often get IP addresses belonging to the hosting ISP. This was the case with the CDN nodes in NORDUnet. NORDUnet have CDN nodes from Akamai, Google and Netflix implemented in their network, and traffic that origin from these were not detected in the first scripts. Additional scripts, filtering based on the CDN node IP prefixes, had to be created and executed. This was done by only filtering out traffic from the three CDN providers, based both on ASN and IP prefixes. The method made sure that traffic from these CDN providers was not counted twice, but at the same time (because of available time) this caused the filtering process to be done in several steps. The first filtering was done for all the chosen CDN providers based on ASN, while the next filtering only included the three CDN providers based on ASN and IP prefixes. In addition, Twitch was first filtered with a wrong ASN, and had to be filtered one more time as the last filtering. The impact have been that results had to be handled more manually than expected, e.g. summing up the total traffic. This may be a source of error, especially because of the asynchronous sampling.

The when and where, is also impacted by the slow start. The plan was to use collected traffic from a week in February, and the goal was to cover several network nodes in Uninett to gain a more complete overview. All the nodes with connections out of Uninett were considered of interest.
Because of the slow start, and the pandemic covid-19, data was collected for March 10th and 11th 2020. After March 12th, which was the day where the Norwegian

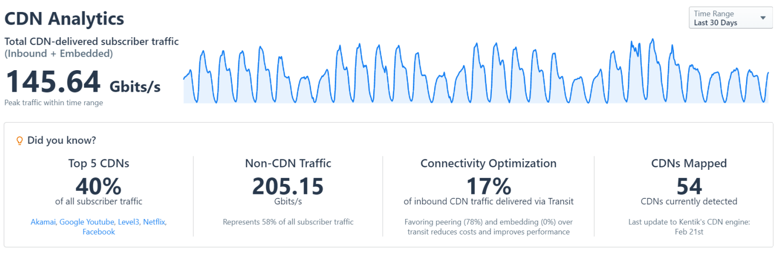
government broadcasted the new guidelines in relation to covid-19 [KS20], the user behaviour changed. Therefore, data before March 12th was needed, and since Uninett stores the full dataset for a limited amount of time, March 10th and 11th were the data sets that could be collected.
The router oslo-gw1 was the node the data set was collected at, and due to project time limitation, this was assessed to give good enough insight in to the CDN traffic. Two of the most important links related to CDN traffic, terminate at oslo-gw1. Another link that would have been relevant is tullin-gw1 - NORDUnet (DK) which have peaks around 25 Gbps [Uni20]. Two additional links goes from Uninett to NORDUnet (SE) (from narvik-gw2 and utsjok-gw1), which should be added to the measurements. Though, the assumption is that these two links would not highly influence the result, since the traffic peaks are around 4 Gbps and 1,5 Gbps [Uni20].

As mentioned, Uninett employees had to execute the scripts. This has made the process of obtaining results demanding, and it is reasonable to believe that the quality of the results has been influenced. This adds an extra source of error, especially in relation to executing the scripts in the correct order, storing the results and using the correct variables. On the other hand, the employee in Uninett has good experience with SiLK, and it may have improved the process.

NORDUnet's set up in Kentik has been used both to aid the choosing of CDN providers and to compare the results with. Information about the configuration of NORDUnet's network in Kentik is limited. Kentik relies on that interfaces are characterized, and that all information about the network has been provided (e.g. implemented CDN nodes). How thorough this set up is, is not known. Kentik does not show any information about NORDUnet having implemented any CDN nodes, which they have, and this shows that the results from Kentik must be critically evaluated. It is important to state that Kentik has been used as inspiration and as a hypothesis of the amount of CDN traffic, not as an explicit answer.

6.2 Results

The assumption was that the amount of CDN traffic in Uninett could be compared to the amount in NORDUnet. Kentik calculates the amount of CDN traffic based on the peak value, and a view from March 17th valid for the "last 30 days" shows that the amount of CDN traffic is around 42% (see Figure 6.1). The results in Uninett is around 20%, if all traffic is considered. It is less than in NORDUnet. The results from Uninett do most likely not contain all the CDN traffic, due to that e.g. the inter-domain link to NORDUnet in Denmark is not measured. By including all the inter-domain links, the amount would probably be higher, but there is no guarantees that it will be as high as in NORDUnet. This project still pose as a good representation of the CDN traffic, due to the assumption that most of the CDN traffic arrives from NORDUnet (SE).



CDN Analytics

Time Range: Last 30 Days

Total CDN-delivered subscriber traffic (Inbound + Embedded)

145.64 Gbits/s

Peak traffic within time range

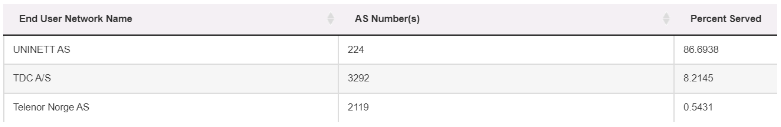
Did you know?

Top 5 CDNs	Non-CDN Traffic	Connectivity Optimization	CDNs Mapped
40%	205.15	17%	54
of all subscriber traffic	Gbits/s	of inbound CDN traffic delivered via Transit	CDNs currently detected
Akamai, Google Youtube, Level3, Netflix, Facebook	Represents 58% of all subscriber traffic	Favoring peering (78%) and embedding (0%) over transit reduces costs and improves performance	Last update to Kentik's CDN engine: Feb 21st

Figure 6.1: CDN Traffic in NORDUnet for the last 30 days from March 17th, 2020 [Kennd]

The results is also lower than the forecast by Cisco (see Figure 2.1). Details about how the measurements in NORDUnet and forecast in Cisco are achieved, are not known. It is most likely different from the method in this project, and may be one reason why the results are different. In addition, it may be lower than Cisco's forecast due to that Uninett is a NREN.

The traffic with destination ASN as Uninett (in) carries over 90% of the all the CDN traffic in Uninett. This is as expected. CDN providers often have a heavy outbound traffic ratio, while the ISP is at the receiving end. Uninett do not transit very much of the CDN traffic to other ISPs; the results from the type "ext2ext" are insignificant. Numbers available in the Akamai Network Portal (see Figure 6.2) states that of all usage, Uninett uses the nodes 86,7%. This supports the expectation that most of the traffic is coming in to Uninett, and that the transit traffic to other ISPs is low.



End User Network Name	AS Number(s)	Percent Served
UNINETT AS	224	86.6938
TDC A/S	3292	8.2145
Telenor Norge AS	2119	0.5431

Figure 6.2: How much of the traffic from the CDN nodes in Uninett serves other ISPs (June 15th, 2020) [Akandb]

The values for all the traffic in Uninett are based on sampled traffic every 2 minute. This is the same as for SiLK. SiLK is set to count every 2 minute (bin size = 120 seconds), and it appears in the results like they do exactly that. When it comes to the values provided by Uninett, the sampling rate is approximately every 2 minutes, but not exact every 2 minutes. This is because the traffic statistics is collected around midnight from one router at the time. This creates small differences in sampling times. From the results it appears like the Uninett sampling starts at around 00:04, and ends at around 23:58. This creates the low starting points in Figure 4.3, since the traffic to and from the Uninett nodes from Akamai, which provides much of the CDN traffic, does not start before 00:04.

All the traffic that is associated with the ASNs and IP prefixes of the CDN providers, and Uninett's CDN node traffic, are characterized as CDN traffic. This may not be only CDN traffic. It may be that some of the traffic can be characterized as other types of traffic, like control traffic. The assumptions is that this compose so little of the characterized CDN traffic, that this is not looked into any further. To get an even more correct picture, this should be considered in the future.

The results showing the graphic representation of the amount of CDN traffic are presented in Figure 4.3. The distribution of CDN traffic is assumed to follow approximately the same distribution as for all traffic. This would in "normal" ISPs give highest amount of CDN traffic in the evening after business hours [FPS+13]. Uninett is a research and education network, so their results are expected to be highest during business hours. March 10th shows an even usage through the day, while March 11th shows most usage during business hours. This is not exact the same as for all traffic (see Figure 4.4), but March 11th looks most like the expected distribution.

The results from the attempt to validate and test the accuracy in SiLK is presented in Table 4.6. The results shows that the sum of CDN traffic, non-CDN traffic (the "fail"-result) and int2int traffic is higher than when all traffic is filtered together. All traffic is defined to be the types in, inweb, out, outweb, int2int and ext2ext. One initial reason for the difference was that the results for Twitch[1] and Dropbox were counted twice. But this is corrected for in Table 4.6, and there still is a significant difference. The reason for this is unknown, and it can be used as a measure of how inaccurate the measurements are. If the peak difference is used, the deviation can be calculated based on the highest peak difference and the average of the values for All SiLK traffic and the sum of CDN, non-CDN and int2int, see Table 6.1. The average difference can also be used to calculate the deviation, and is presented in Table 6.2. Values are in Gbps.

6.2.1 Source of error

As already mentioned, the traffic on all the inter-domain links are not measured. Therefore the actual CDN traffic is probably higher.
The measurement period is only two days. This can lead to that abnormal situations

[1]The first results, which was the incorrect and where the traffic was close to zero

Table 6.1: Deviation based on peak difference in SiLK

Date	**CDN+nonCDN +int2int**	**All SiLK traffic**	**Avg.**	**Peak Difference**	**Deviation**
March 10th	46,26	44,17	45,22	2,09	4,6%
March 11th	18,53	17,12	17,82	1,41	7,9%

Table 6.2: Deviation based on average difference in SiLK

Date	**CDN+nonCDN +int2int**	**All SiLK traffic**	**Avg.**	**Avg. Difference**	**Deviation**
March 10th	28,16	27,65	27,91	0,51	1,8%
March 11th	29,96	29,71	29,84	0,26	0,9%

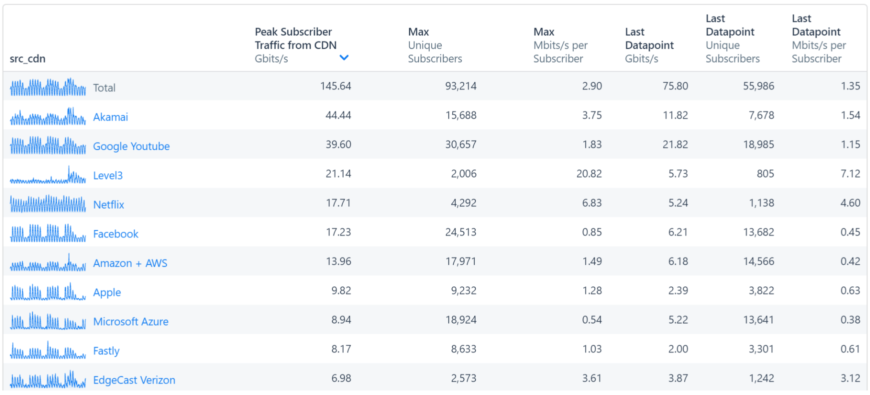
are missed, or it may be abnormal in the measurement period. A longer period, e.g. a month, should be considered to make sure that the normal situation is detected.
The execution of the scripts was done by Uninett employees, which can cause errors due to e.g. not fully understanding the scripts, the execution order and the sorting of files and folders.
A lot of the preparations before executing scripts (e.g. making the scripts) and analysing the results are done manually. This can cause errors due to human mistakes. This happened with the filtering of CDN traffic for Twitch and the CDN caches in NORDUnet, and led to unnecessary manual analysis.
To achieve peak-results for the CDN traffic, the peak samples where added to each other. This is not completely correct, since the sampling rate is not synchronized.
The SiLK Flow records are converted from the records collected through IPFIX. IPFIX samples every record (1:1), which can lead to buffer overflow and flows collection failure. This may cause the result in this project to be lower than it actual is.

6.3 Recommended model

The recommended model is based on several criteria. These criteria influence the outcome, and other criteria can give other recommendation. The criteria that is considered the most important in this project is to decrease the load on central links, and to handle abnormal situations (like flash crowds).

The economical evaluation is based on a proposed model of different costs. If the solutions are economical beneficial may be highly dependent of the traffic load, which should be considered further. It is difficult to assess if the traffic load is high enough, without having full overview of the costs. The developed economical model can be used by an ISP to aid this decision. The cost of upgrading links, and the possible saved cost by postponing upgrading, should also be considered further. This article [LMP12] points to that the amount of traffic, the congestion, have to be high enough before implementing a CDN node is beneficial.

It is assumed that the peering and CDN node suggestions in the recommended model will not affect NORDUnet. Figure 6.3 shows the traffic peaks for the biggest CDN providers in NORDUnet from Kentik[2] [Kennd]. If assumed that all the traffic from the CDN providers came through the NORDUnet(SE)-link (excluding CDN providers with peering agreements via NIX), then e.g. Uninett's traffic from/to Google represents around 7% [3] and from/to Netflix represents around 11%[4] of the NORDUnet traffic. Because the time period for this project's results is two days and one month for the Kentik results, it may be that the peak results would have been higher in a months view. But based on the results from this project, it may be assumed that moving some traffic away from the link to NORDUnet(SE) will not degrade NORDUnet's services.



src_cdn	Peak Subscriber Traffic from CDN Gbits/s	Max Unique Subscribers	Max Mbits/s per Subscriber	Last Datapoint Gbits/s	Last Datapoint Unique Subscribers	Last Datapoint Mbits/s per Subscriber
Total	145.64	93,214	2.90	75.80	55,986	1.35
Akamai	44.44	15,688	3.75	11.82	7,678	1.54
Google Youtube	39.60	30,657	1.83	21.82	18,985	1.15
Level3	21.14	2,006	20.82	5.73	805	7.12
Netflix	17.71	4,292	6.83	5.24	1,138	4.60
Facebook	17.23	24,513	0.85	6.21	13,682	0.45
Amazon + AWS	13.96	17,971	1.49	6.18	14,566	0.42
Apple	9.82	9,232	1.28	2.39	3,822	0.63
Microsoft Azure	8.94	18,924	0.54	5.22	13,641	0.38
Fastly	8.17	8,633	1.03	2.00	3,301	0.61
EdgeCast Verizon	6.98	2,573	3.61	3.87	1,242	3.12

Figure 6.3: Some statistics for CDN providers in NORDUnet [Kennd]

Both Twitch and Fastly are two of the 14 big CDN providers, and they are available for peering in Oslo, but the recommended model does not suggest that peering agreements should be established. There are two separate reasons for this. The information found about Twitch is limited, and it is difficult to assess if peering could be beneficial. The reason why it is not recommended, is because NORDUnet has a peering-agreement with Twitch. Based on Figure 6.3, Twitch is not listed as the biggest and it can be assumed that the numbers for Twitch are below 6,98 Gbps.

[2]"Last month"-view from March 17th, 2020
[3]2,69 Gbps / 39,60 Gbps = 0,068
[4]1,97 Gbps / 17,71 Gbps = 0,111

”Worst case”, Uninett may be responsible for minimum 34%[5] of the traffic. This may be too high, and may impact NORDUnet’s services.
Fastly are available for private peering at Digiplex, but because the traffic peak are measured to be 1,47 Gbps, it is assessed too low if economical costs are compared to the advantages [Eid20].

Given the situation where all the Nordic NRENs implement solutions like recommended to Uninett, it may affect NORDUnets ability to provide a good service. Implementing CDN solutions should be seen in coherence with the traffic volume in NORDUnet, like with Twitch where the recommendation to Uninett was to not establish their own agreement. This is the immediate situation, while it may change in the future. Assuming that the traffic volume will increase, then it may be beneficial for the other NRENs and for NORDUnet that they implement CDN solutions.

There may be some downside aspects with CDN nodes that the ISP should be aware of. The ISP can not control the content on the CDN nodes, and have to rely on that the CDN provider stores relevant content on the node. The cost of the objects requested by users may therefore still apply the same costs on the ISP (e.g. transit), if the objects are not stored at the CDN node. Which may be a drawback for the ISP if the objects are popular [ARM16].

6.3.1 Expert-meeting

Validation of the model was done through an expert meeting. The main outcomes from this meeting was that the criteria selection may have been done otherwise, where performance should have been considered. The economical aspects with the model was discussed more than anticipated. Because of this, the economical aspects was looked into more than originally planned. The expert group agreed on the model being relevant considering the criteria.

6.4 Ethics and privacy concerns

Net neutrality is a discussion when it comes to CDN and their collaboration with ISPs [Lea14]:

It is possible to question if it is fair to other content providers, if the ISP accepts CDN nodes from specific content providers but not from others. Especially, if such a node offers extra services available only to the customers of the ISP.

Another question that may be raised is if peering between ISPs and CDNs can be a challenge. The peering itself is not a problem, but it can be a problem if ISPs allows peering with some CDN providers, but refuse peering with other. This is not fair, and is in conflict with the net neutrality principle.

[5]2,37 Gbps / 6,98 Gbps = 0,339

The following paragraph is from the pre-project of this master thesis [Asp19], and ist still valid and of big importance in this project:

Protecting the privacy of the users is important due to trust between them and the ISP, but privacy is also defined according to Law. Both the Norwegian Constitutional Law §102 and European Convention on Human Rights article 8 states that people have right to privacy, and that communication also should be protected [sosI15].

In this project the privacy of the users is protected by only getting access to data containing no sensitive information. The full data set have not been available to other than employees in Uninett, and the results only withheld data about the traffic load, not any data that can be traced back to specific users.

6.5 Alternative solutions

The different solutions presented in Section 5.5 was Smart Control Plane (SCP), Licensed CDN, MicroCDN, lightweight in-network caching and I-CDN. These solutions aims to make the utilization of the ISP network better, and the content delivery should be optimized.

SCP seems like a promising alternative, based on that it will handle the management of the ICN-part on top of already existing IP network infrastructure. SCP can also control the cooperation with other ISPs, and have both a positive intra- and interdomain effect [STUA18]. The alternative is presented with the need of a central high performance server to handle the management, which may be a disadvantage. The possibility to distribute the management can be a better solution, which also is more robust.
Licensed CDN relies on the ISP paying a license for the CDN solutions provided by the CDN provider [HNC+16]. This cost may not be beneficial for the ISP, and this solution does not necessary give the ISP control over the aspects that in-network caches may give.
In-network cache solutions will give the ISP more control over their own network, and make it easier to optimize the content delivery. There are different approaches, like I-CDN [KMK+13] and lightweight caching [TCLP13]. Which alternative that fits the ISP best depends on e.g. the size of the content.
MicroCDN utilizes in-network caching, but also aims at using CCN to solve the content delivery [IMR14]. By using CCN, different workarounds that are made to overcome challenges in content delivery, can be condemned unnecessary.

All these solutions may impact the traffic pattern and volume in the ISP, which lead to a change in the economical cost. Introducing e.g. CCN may reduce the costs for a Tier 2 or 3 ISP [Kam15]. Before implementing any of the alternative solutions, costs should be assessed further.

The different solutions will inflict changes in Uninett. Some of the solutions will demand big changes and new thinking, like SCP which will imply adding a new layer to manage ICN.

The solutions may improve the handling of CDN traffic, even though Uninett is the only one implementing them. The key is to cache content inside Uninett and keep track of where the content is and how it should be delivered to users. It may decrease the need of content delivery from inter-domain links, and give Uninett the control. This will cause additional work to Uninett, but if it is cost-efficient it can be beneficial, and should be looked into further.

Chapter 7 Conclusion

This Chapter concludes the work in this master thesis, and contains the conclusion and recommended further work. The solution to the main objective is provided through a recommended model, while the answers to the research questions have been used in the process of creating the model.

7.1 Conclusion

The important research questions in this project have been how much CDN traffic Uninett contains, and how this can be measured. The answer to these questions, together with the criteria, were the baseline for developing a recommended model for how CDN technologies could be used in Uninett.

Finding a method how CDN traffic could be measured was achieved by using the tool SiLK and filter the traffic based on ASN. Different checks were implemented to accomplish a good list with relevant ASNs of 40 CDN providers. The implementation of Kentik in NORDUnet has been used as an inspiration, guideline and basis for comparison. The measurement process posted as challenging, because of two main factors. The data set had to be privacy protected, and all the scripts that filtered the traffic had to be executed by Uninett employees. Secondly, during the project period, the pandemic Covid-19 changed the traffic behaviour. These two factors may have impacted the final results.
In addition to using SiLK, some raw data from Uninett's own traffic statistics has been used to achieve the final result.

The final result of how much CDN traffic it is in Uninett is calculated for the average and peak values. It has also been relevant to calculate the amount of CDN traffic in to Uninett. Both for average and peak traffic values, the total amount of CDN traffic in Uninett is around 20%. If only the direction ”in” is considered, the amount is over 50%. Both results are lower than the ones in NORDUnet (based on Kentik) and the prediction from Cisco. This may be because not all the relevant links in Uninett are measured, or that the measurement and calulation method used, are different from those used by NORDUnet (Kentik) and Cisco.

The peak values for the CDN providers have been used to determine which CDN providers it will be relevant to collaborate with, and by collaborating optimizing the use of CDN technologies. The chosen CDN technologies are peering or implementing CDN nodes in the ISP network, and are chosen based on being realistic to implement. The main criteria have been to move traffic away from central inter-domain links to postpone upgrading, and to better be able to handle flash crowds. The consideration of the economical aspects have been challenging, and needs more work.

The recommended model made for Uninett is to host CDN nodes from Akamai and Google Global Cache, publicly peer with Cloudflare and Dropbox and privately peer with Microsoft and Amazon.

This model was validated through an expert meeting, and the outcome was that based on the criteria set, it could contribute to optimizing the use of CDN technologies in Uninett.

The measurement method can be adapted to other ISP scenarios, and based on their own results they can make use of the model. The overview of CDN providers and requirements for collaboration can also be of use for other ISPs.

Future technologies like ICN, or alternatives like in-network caching, can also be used to optimize content delivery. Such solutions may be added on top of the IP network, which makes it more approachable, and should be considered further.

7.2 Further work

The measurement method in this project is passive, and performance is not considered in this project. Performance should be looked into, to better be able to choose which CDN providers it is relevant to collaborate with. This was also brought up during the expert meeting. Active measurements of the relevant CDN providers should be conducted, e.g. in similar ways like Emir Beganović [Beg19]. Results for the performance will also provide a better basis for establishing agreements with CDN providers.

The work done in this project can be improved and automated, so that measurements of the CDN traffic can be done as a routine. Today the traffic volume requirements of some CDN providers exclude Uninett from collaborating. By having updated results of the amount of CDN traffic, it will be easier to see if Uninett can meet the requirements.

NORDUnet has already implemented CDN nodes and peering agreements, but by developing the measurement method in this project even more, Uninett can possibly advice NORDUnet which agreements to have. If performance is added to the method, it will be even more relevant.

The approach considered in the beginning of this project, by using recursive DNS requests, should be looked further into. The results in this project may exclude measurements of CDN traffic that origins from CDN nodes in other networks than Uninett and NORDUnet. Recursive DNS may provide results including that type of CDN traffic.

The model can be developed further, and formal expressions can be created to make the model easier to use for other ISPs. The goal should be to create a model where ISPs can insert their own values, resulting in a customized recommended solution.

Future technologies and other alternatives to optimize the content delivery in an ISP should be looked further into. Especially Smart Control Plane (SCP) and in-network caching should be considered further.

References

[AJ00] Martin Arlitt and Tai Jin. A workload characterization study of the 1998 world cup web site. *IEEE Network*, 14(3):30–37, 2000.

[Akanda] Akamai. Akamai network partnerships, n.d. Available at: https://www.akamai.com/us/en/products/network-operator/akamai-network-partnerships.jsp. Last accessed: May 23rd, 2020.

[Akandb] Akamai. Aura operator portal, n.d. Available through: https://network.akamai.com/portal/. Last accessed: June 2nd, 2020.

[Akandc] Akamai. Facts and figures, n.d. Available at: https://www.akamai.com/us/en/about/facts-figures.jsp. Last accessed: May 24th, 2020.

[Akandd] Akamai. What does cdn stand for? cdn definition, n.d. Available at: https://www.akamai.com/us/en/cdn/what-is-a-cdn.jsp. Last accessed: May 13th, 2020.

[Amand] Amazon. Amazon cloudfront key features, n.d. Accessed at: https://aws.amazon.com/cloudfront/features/?nc=sn&loc=2. Last accessed: June 19th, 2020.

[APN] APNIC. Geoff Huston. Available at: https://www.apnic.net/about-apnic/team/geoff-huston/. Last accessed: June 18th, 2020.

[Appnd] Apple. Apple edge cache, n.d. Available at: https://cache.edge.apple/. Last accessed: June 18th, 2020.

[ARM16] Andrea Araldo, Dario Rossi, and Fabio Martignon. Cost-aware caching: Caching more (costly items) for less (isps operational expenditures). *IEEE Transactions on Parallel and Distributed Systems*, 27(5):1316–1330, 2016.

[Asp19] Sissel-Johanne Aspdal. CDN in Uninett. Project report in TTM4502, Department of Information Security and Communication Technology, NTNU – Norwegian University of Science and Technology, Dec. 2019.

[Asp20] Sissel-Johanne Aspdal. CDN Measurements, 2020. Available at: https://github.com/sisseljohanne/cdnmeasurements.

[Bai10] Craig Bailey. Content is king by bill gates, 2010. Available at: https://www.craigbailey.net/content-is-king-by-bill-gates/. Last accessed: June 9th, 2020.

[BBDN17] Halvor Bothner-By, Henrik Dvergsdal, and Ola Nordal. Arpanet, 2017. Available at: https://snl.no/ARPANET. Last accessed: May 23rd, 2020.

[Beg19] Emir Beganović. Analysing global cdn performance, August 2019. Available at: https://labs.ripe.net/Members/emirb/analysing-global-cdn-performance. Last accessed: November 14th, 2019.

[Bha19] Anurag Bhatia. Facebook fna node update, 2019. Available at: https://anuragbhatia.com/2019/11/networking/isp-column/facebook-fna-node-update/. Last accessed: June 19th, 2020.

[BPV08] Rajkumar Buyya, Mukaddim Pathan, and Athena Vakali. *Content Delivery Networks*, volume 9 of *Lecture Notes in Electrical Engineering*. Springer Berlin Heidelberg, 1st edition, 2008.

[BS16] Pierre-Jean Benghozi and Jean-Paul Simon. Out of the blue: the rise of cdn networks.(extra paper). *DigiWorld Economic Journal Communications Strategies*, 101:107, 2016.

[CAoCS18] Chinese Academy Chinese Academy of Cyberspace Studies. *World Internet Development Report 2017: Translated by Peng Ping*. Springer, Berlin, Heidelberg, 1st edition, 2018.

[CAR18] Shujie Cui, Muhammad Rizwan Asghar, and Giovanni Russello. Multi-cdn: Towards privacy in content delivery networks. *IEEE Transactions on Dependable and Secure Computing*, 2018.

[CCL18] Claudia Canali, Andrea Corbelli, and Riccardo Lancellotti. Designing a private cdn with an off-sourced network infrastructure: model and case study. In *2018 26th International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, pages 1–6. IEEE, 2018.

[CGF$^+$15] Danilo Cicalese, Danilo Giordano, Alessandro Finamore, Marco Mellia, Maurizio Munafò, Dario Rossi, and Diana Joumblatt. A first look at anycast cdn traffic. *arXiv preprint arXiv:1505.00946*, 2015.

[CGLLP12] Fangfei Chen, Katherine Guo, John Lin, and Thomas La Porta. Intra-cloud lightning: Building cdns in the cloud. In *Proceedings - IEEE INFOCOM*, pages 433–441, 2012.

[Cis12] Cisco. Introduction to cisco ios netflow - a technical overview, 2012. Available at: https://www.cisco.com/c/en/us/products/collateral/ios-nx-os-software/ios-netflow/prod_white_paper0900aecd80406232.html. Last accessed: November 17th, 2019.

[Cis18] Cisco. Cisco visual networking index: Forecast and trends, 2017–2022 white paper. Technical Report C11-741490-00, Cisco, November 2018.

[Cisnd] Cisco. Content distribution overview, n.d. Available at: https://www.cisco.com/c/dam/global/it_it/solutions/pdf/ipcom/cisco-cdn.pdf. Last Accessed: October 27th, 2019.

[Clo] Cloudflare. Cloudflare peering policy. Available at: https://www.cloudflare.com/peering-policy/. Last accessed: May 27th, 2020.

[Clond] Cloudflare. The cloudflare global anycast networks, n.d. Available at: https://www.cloudflare.com/network/. Last accessed: May 27th, 2020.

[Dro] Dropbox. Peering. Available at: https://www.dropbox.com/peering. Last accessed: May 27th, 2020.

[Drond] Dropbox. Our story, n.d. Available at: https://www.dropbox.com/about. Last accessed: May 27th, 2020.

[DSA$^+$11] Florin Dobrian, Vyas Sekar, Asad Awan, Ion Stoica, Dilip Joseph, Aditya Ganjam, Jibin Zhan, and Hui Zhang. Understanding the impact of video quality on user engagement. *ACM SIGCOMM Computer Communication Review*, 41(4):362–373, 2011.

[DVR11] Danny De Vleeschauwer and Dave C. Robinson. Optimum caching strategies for a telco cdn. *Bell Labs Technical Journal*, 16(2):115–132, 2011.

[Eid20] Håvard Eidnes. Personal communication. Senior Consultant Uninett, 2019-2020.

[Ell20] Silje Marnburg Ellefsen. Uninett kobler forskningsnettet direkte til microsofts datasenter i norge, 2020. Available at: https://www.uninett.no/uninett-kobler-forskningsnettet-direkte-til-microsofts-datasenter-i-norge. Last accessed: June 11th, 2020.

[Excnd] Norwegian Internet Exchange. Pricing, n.d. Available at: https://www.nix.no/pricing/. Last accessed: June 8th, 2020.

[FN] FN. Internettbrukere. 2018. Available at: https://www.fn.no/Statistikk/Internettbrukere. Last accessed: May 23rd, 2020.

[For13] Internet Engineering Task Force. Specification of the ip flow information export (ipfix) protocol for the exchange of flow information, 2013. Available at: https://tools.ietf.org/html/rfc7011. Last accessed: November 17th, 2019.

[FPL$^+$13] Benjamin Frank, Ingmar Poese, Yin Lin, Georgios Smaragdakis, Anja Feldmann, Bruce Maggs, Jannis Rake, Steve Uhlig, and Rick Weber. Pushing cdn-isp collaboration to the limit. *ACM SIGCOMM Computer Communication Review*, 43(3):34–44, 2013.

[FPS$^+$13] Benjamin Frank, Ingmar Poese, Georgios Smaragdakis, Anja Feldmann, Bruce Maggs Maggs, Steve Uhlig, Vinay Aggarwal, and Fabian Schneider. Collaboration opportunities for content delivery and network infrastructures. *Recent Advances in Networking*, 1:305–377, 2013.

[Goo] Google. Edge nodes (ggc). Available at: https://peering.google.com/#/options/google-global-cache. Last accessed: June 18th, 2020.

[goo19] How google is building the fastest internet cable to cross the atlantic. *ICT Monitor Worldwide*, 2019.

[Goonda] Google. Cloud cdn, n.d. Available at: https://cloud.google.com/cdn. Last accessed: May 15th, 2020.

[Goondb] Google. Introduction to ggc, n.d. Available at: https://support.google.com/interconnect/answer/9058809?hl=en. Last accessed: May 23rd, 2020.

[Goondc] Google. Our infrastructure, n.d. Available at: https://peering.google.com//infrastructure. Last accessed: May 24th, 2020.

[Goondd] Google. Peering, n.d. Available at. https://peering.google.com/#/options/peering. Last accessed: June 18th, 2020.

[HNC$^+$16] Nicolas Herbaut, Daniel Negru, Yiping Chen, Pantelis A. Frangoudis, and Adlen Ksentini. Content delivery networks as a virtual network function: A win-win isp-cdn collaboration. In *2016 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6, 2016.

[Hor20] Angela Horneman. Personal communication. Situational Awareness Analysis Team Lead, 2020.

[Husnd] Geoff Huston. Autonomous system number-to-name, n.d. Available at: https://bgp.potaroo.net/cidr/autnums.html. Last accessed: June 5th, 2020.

[HWC$^+$16] Han Hu, Yonggang Wen, Tat-Seng Chua, Jian Huang, Wenwu Zhu, and Xuelong Li. Joint content replication and request routing for social video distribution over cloud cdn: A community clustering method. *IEEE Transactions on Circuits and Systems for Video Technology*, 26(7):1320–1333, 2016.

[IMR14] Claudio Imbrenda, Luca Muscariello, and Dario Rossi. Analyzing cacheable traffic in isp access networks for micro cdn applications via content-centric networking. In *Proceedings of the 1st ACM Conference on Information-Centric Networking*, pages 57–66, 2014.

[Insnd] Carnegie Mellon University Software Engineering Institute. Route views project page, n.d. Available at: https://routeviews-mirror.cert.org/. Last accessed: June 5th, 2020.

[Jan12] Grzegorz Janoszka. Peering basics: Public vs. private peering, 2012. Available at: https://blog.leaseweb.com/2012/10/24/public-vs-private-peering-the-basics-part-1/. Last accessed: June 19th, 2020.

[Kam15] Noriaki Kamiyama. Analyzing impact of introducing ccn on profit of isps. *IEEE transactions on network and service management*, 12(2):176–187, 2015.

[Ken] Kentik. Network traffic analysis. Available at: https://www.kentik.com/solutions/network-monitoring-and-analytics-for-service-providers. Last accessed: May 16th, 2020.

[Kennd] Kentik. Kentik portal, n.d. Available at: https://portal.kentik.eu/. Last accessed: May 24th, 2020.

[KMK$^+$13] Noriaki Kamiyama, Tatsuya Mori, Ryoichi Kawahara, Shigeaki Harada, and Haruhisa Hasegawa. Analyzing influence of network topology on designing isp-operated cdn. *Telecommunication Systems*, 52(2):969–977, 2013.

[Kom18] Kompetanse Norge. Befolkningens bruk av digitale verktøy, 2018. Available at: https://www.kompetansenorge.no/statistikk-og-analyse/grunnleggende-digital-ferdigheter/befolkingens-bruk-av-digitale-verktoy/. Last accessed: May 23rd, 2020.

[KOSS19] Paul Krystosek, Nancy M. Ott, Geoffrey Sanders, and Timothy Shimeall. *Network Traffic Analysis with SiLK. Analyst's Handbook for SiLK Versions 3.15.0 and Later.* Carnegie Mellon Uninversity, June 2019.

[KP18] Mehdi Khosrow-Pour. *Encyclopedia of Information Science and Technology, Fourth Edition.* Information Science Reference, 2018.

[KS20] Pedja Kalajdzic and Emilie Louise Solberg. Varsler de mest inngripende tiltakene norge har hatt i fredstid, 2020. Available at: https://www.nrk.no/norge/varsler-de-mest-inngripende-tiltakene-norge-har-hatt-i-fredstid-1.14940376. Last accessed: June 19th, 2020.

[KWS12] Bill Krogfoss, Marcus Weldon, and Lev Sofman. Internet architecture evolution and the complex economies of content peering. *Bell Labs Technical Journal*, 17(1):163–184, 2012.

[Lea14] Maria Cristina Leal. The eu approach to net neutrality: Network operators and over-the-top players, friends or foes? *Computer law & security review*, 30(5):506–520, 2014.

[LMP12] Dongmyung Lee, Jeonghoon Mo, and Jinwoo Park. Isp vs. isp+ cdn: Can isps in duopoly profit by introducing cdn services? *ACM SIGMETRICS Performance Evaluation Review*, 40(2):46–48, 2012.

[LS13] Zhe Li and Gwendal Simon. In a telco-cdn, pushing content makes sense. *IEEE Transactions on Network and Service Management*, 10(3):300–311, 2013.

[MA16] Jason McGee-Abe. Tech giants to build 160tbps transatlantic cable. *Global Telecoms Business*, 2016.

[Mar19] IDC Marketscape. Idc marketscape: Worldwide commercial cdn 2019 vendor assessment, August 2019. Available at: https://www.centurylink.com/asset/business/enterprise/report/idc-marketscape-worldwide-commercial-cdn-2019-vendor-assessment-report.pdf. Last accessed: June 6th, 2020.

[Mic19] Microsoft. Peering policy, 2019. Available at: https://docs.microsoft.com/nb-no/azure/internet-peering/policy. Last accessed: June 5th, 2020.

[Mic20] Microsoft. Azure cdn coverage by metro, 2020. Available at: https://docs.microsoft.com/en-us/azure/cdn/cdn-pop-locations. Last accessed: June 5th, 2020.

[Micnd] Microsoft. Microsoft edge caching program, n.d. Available at: https://peering.azurewebsites.net/Peering/Caching. Last accessed: June 5th, 2020.

[Netnda] Netflix. Open connect, n.d. Available at: https://openconnect.netflix.com/en. Last accessed: May 23rd, 2020.

[Netndb] Netflix. Requirements for deploying embedded appliances, n.d. Available at: https://openconnect.zendesk.com/hc/en-us/articles/360034538352. Last accessed: May 23rd, 2020.

[Netndc] CERT NetSA. Silk, n.d. Available at: https://tools.netsa.cert.org/silk/. Last accessed: November 17th, 2019.

[Nor10] William B. Norton. A business case for peering in 2010. Technical report, 2010. Available at http://drpeering.net/white-papers/A-Business-Case-For-Peering.php. Last accessed: June 8th, 2020.

[NSS10] Erik Nygren, Ramesh K. Sitaraman, and Jennifer Sun. The akamai network: A platform for high-performance internet applications. *SIGOPS Oper. Syst. Rev.*, 44(3):2–19, August 2010.

[PB07] Al-Mukaddim Khan Pathan and Rajkumar Buyya. A taxonomy and survey of content delivery networks. Technical report, Grid Computing and Distributed Systems (GRIDS) Laboratory, University of Melbourne, 2007.

[PB12] Louis Plissonneau and Ernst Biersack. A longitudinal view of http video streaming performance. In *Proceedings of the 3rd Multimedia Systems Conference*, pages 203–214, 2012.

[Peend] PeeringDB. The interconnection database, n.d. Available at: https://www.peeringdb.com/. Last accessed: June 5th, 2020.

[PFS$^+$12] Ingmar Poese, Benjamin Frank, Georgios Smaragdakis, Steve Uhlig, Anja Feldmann, and Bruce Maggs. Enabling content-aware traffic engineering. *ACM SIGCOMM Computer Communication Review*, 42(5):21–28, 2012.

[pri17] The future of cdn: Hybrid federation models to augment private infrastructure.(conversant connecting digital world). *Streaming Media*, page 73, 2017. Available at: https://search.proquest.com/docview/1891317935?rfr_id=info%3Axri%2Fsid%3Aprimo. Last accessed: June 5th, 2020.

[PSR14] Mukaddim Pathan, Ramesh K. Sitaraman, and Dom Robinson. *Advanced content delivery, streaming, and cloud services.* Wiley series on parallel and distributed computing. Wiley, 2014.

[Rob17] Dom Robinson. *Content Delivery Networks - Fundamentals, design and evolution.* Wiley, 2017.

[Sch15] Tom Scholl. Internet routing and traffic engineering, 2015. Available at: https://aws.amazon.com/blogs/architecture/internet-routing-and-traffic-engineering/. Last accessed: June 19th, 2020.

[sosI15] Departementenes sikkerhets-og serviceorganisasjon Informasjonsforvaltning. Nou 2015:13. digital sårbarhet – sikkert samfunn. 2015.

[Standa] Stackpath. Network, n.d. Available at: https://www.stackpath.com/why-stackpath/network/. Last accessed: June 17th, 2020.

[Standb] Stackpath. Peering policy, n.d. Available at: https://www.stackpath.com/legal/peering-policy/. Last accessed: June 17th, 2020.

[Str20] Marty Strong. Personal communication. Edge Strategy Manager Facebook, 2020.

[STUA18] Mahamah Sebakor, Nipon Theera-Umpon, and Sansanee Auephanwiriyakul. Smart control plane for information centric network-internet service provider networks. *Journal of Central South University*, 25(10):2410–2422, 2018.

[Sum15] Arielle Sumits. The history and future of internet traffic, 2015. Available at: https://blogs.cisco.com/sp/the-history-and-future-of-internet-traffic. Last accessed: May 23rd, 2020.

[SVS13] Abhigyan Sharma, Arun Venkataramani, and Ramesh K Sitaraman. Distributing content simplifies isp traffic engineering. *ACM SIGMETRICS Performance Evaluation Review*, 41(1):229–242, 2013.

[TCLP13] Daphne Tuncer, Marinos Charalambides, Raul Landa, and George Pavlou. More control over network resources: An isp caching perspective. In *Proceedings of the 9th International Conference on Network and Service Management (CNSM 2013)*, pages 26–33. IEEE, 2013.

[Telnd] Telenor. Cdn, n.d. Available at: https://www.telenorwholesale.no/produkter/cdn/. Last accessed: May 23rd, 2020.

[TP19] Shubhika Taneja and Xavid Pretzer. Google cloud networking in depth: Understanding network service tiers, 2019. Available at: https://cloud.google.com/blog/products/networking/google-cloud-networking-in-depth-understanding-network-service-tiers. Last accessed: May 24th, 2020.

[Uni20] Uninett. Uninett trafikkstatistikk, 2020. Available at: http://stats.uninett.no/. Last accessed: June 19th, 2020.

[Uninda] Uninett. About uninett, n.d. Available at: https://www.uninett.no/en/about-uninett. Last accessed: May 15th, 2020.

[Unindb] Uninett. Forskningsnettet – nettet i kunnskaps-norge, n.d. Available at: https://www.uninett.no/forskningsnettet. Last accessed: May 15th, 2020.

[Unindc] Uninett. Grafisk profil uninett as, n.d. Available at: https://www.uninett.no/grafisk-profil-uninett. Last accessed: May 15th, 2020.

[Unindd] Uninett. Internasjonalt samarbeid, n.d. Available at: https://www.uninett.no/internasjonalt-samarbeid. Last accessed: May 15th, 2020.

[Ver02] Dinesh C. Verma. *Content Distribution Networks: An Engineering Approach.* John Wiley sons, Inc., New York, USA, 2002.

[Vernda] Verizon. Customers, n.d. Available at: https://www.verizondigitalmedia.com/customers/. Last accessed: May 27th, 2020.

[Verndb] Verizon. Network, n.d. Available at: https://www.verizondigitalmedia.com/media-platform/delivery/network/. Last accessed: May 27th, 2020.

[Wie14] Roel J. Wieringa. *Design Science Methodology for Information Systems and Software Engineering.* Springer, Berlin, 2014.

Appendix A

List of ASNs

Overview of ASNs associated with the different CDN providers. The ASNs in bold are present in the pmap-files and are used in the filtering. The ASNs in italicized are added to the overview from Kentik.

MAIN

Akamai

AS12222

AS16625

AS16702

AS17334

AS18680

AS18717

AS20189

AS20940

AS22207

AS23455

AS23903

AS24319

AS30675

AS31107

AS31108

AS31109

AS31110

AS31377

AS32787

AS33905

AS34164

AS34850

AS35204

AS35993

AS35994

AS36183

AS393234

AS393560

AS39836

AS43639

AS45700

AS48163

AS49846

AS55770

AS17204

AS21342

AS21357

AS21399

AS22452

AS23454

AS26008

AS55409

Amazon

AS16509

AS14618

AS55960

AS135629

AS8987

AS19047

AS9059

AS7224

Apple

AS6185

AS714

Cloudflare

AS132892

AS13335

AS133877

AS14789	**AS19527**	**AS560**	AS22692
AS202623	AS22577	Limelight	**AS23468**
AS203898	AS22859	AS12411	AS25796
AS394536	**AS24424**	**AS22822**	AS26222
AS395747	AS26684	**AS23059**	AS30135
AS139242	AS36039	AS23135	AS30575
AS209242	**AS36040**	AS23164	AS31792
Dropbox	**AS36384**	**AS25804**	AS32476
AS19679	**AS36385**	AS26506	**AS35106**
AS200499	**AS36492**	AS27191	**AS3598**
AS203719	AS36561	**AS37277**	AS36006
AS393874	AS36987	**AS38621**	AS395496
AS54372	AS394507	**AS38622**	AS395524
AS62190	AS394639	**AS45396**	AS395851
Facebook	AS394699	**AS55429**	AS396463
AS32934	AS395973	**AS60261**	AS397466
AS54115	AS40873	Microsoft Azure	AS40066
AS63293	**AS41264**	**AS12076**	**AS45139**
Fastly	**AS43515**	**AS13399**	**AS5761**
AS394192	*AS11344*	AS13811	**AS58862**
AS54113	Level3	AS14719	**AS59067**
Google	**AS3356**	AS17345	**AS6182**
AS15169	**AS3549**	AS20046	**AS6194**
AS16550	**AS12576**	AS200517	**AS6291**

AS63314	AS8074	<u>Stackpath</u>	*AS199156*
AS6584	**AS8075**	**AS11588**	***AS64259***
AS8068	**AS8812**	**AS12989**	*AS31936*
AS8069	<u>Netflix</u>	**AS18607**	<u>Twitch</u>
AS8070	**AS2906**	**AS20446**	AS397153
AS8071	**AS40027**	AS29798	**AS46489**
AS8072	<u>Reflected networks</u>	**AS33438**	<u>Verizon's Edgecast</u>
AS8073	**AS29789**	AS54104	**AS15133**

SMALL

Alibaba Cloud

AS24429

Azion

AS52580

BelugaCDN

AS23393

BunnyCDN

AS200325

CacheFly

AS30081

CDNetworks

AS36408

AS38107

AS38670

AS40366

AS43303

cdnnow

AS21030

CDNvideo

AS57363

CDN77

AS60068

ChinaCache

AS63541

G-core CDN

AS199524

AS202422

AS59245

Imperva Incapsula

AS19551

AS62571

Instartlogic

AS133103

AS33047

Internap

AS11855

AS12179

AS12180

AS12182

AS13789

AS13792

AS14742

AS14744

AS14745

AS15570

AS17675

AS19024

AS24245

AS24246

AS24247

AS30282

AS48910

Kingsoft Cloud

AS137280

AS59019

LeaseWeb CDN

AS60626

Medianova

AS21245

Ngenix

AS34789

Pandora

AS22518

AS40428

QUANTIL China NetCenter

AS54994

Tata Communications

AS40009

Tencent Cloud

AS132591

AS133478

AS58835

Yahoo

AS10310

AS14776

AS14777

AS14778

AS14779

AS14780

AS14781

AS23879

AS24506

AS26085

AS26101

AS34010

AS36088

AS36229

AS36646

AS36647

AS56173

AS5779

AS7233

Zenedge

AS393676

Appendix B
Script for the types in and inweb

```
#rwfilterin2020031X.sh
DATA= <PATH TO DATA-SET>
SILKCONF= <PATH TO SILK.CONF>
START=2020/03/11
END=2020/03/11
pmapfile=20200311.bgp.ripe.pmap
BIN=120
#Adapt ASN to the pmap-file used
#.csv-fil: --no-col --column-separator=";"
echo Starts filtering CDN-traffic: in, inweb. Separate
   results for Facebook, Netflix, Akamai, Google Youtube,
    Apple etc.
rwfilter --type=in,inweb --start=$START --end=$END --pmap
   -file=FACEBOOK:$pmapfile --pmap-src-FACEBOOK=AS32934,
   AS54115,AS63293 --pass=passinfb.rw --fail=stdout --
   data=$DATA | rwfilter --type=in,inweb --site-config-
   file=$SILKCONF --pmap-file=NETFLIX:$pmapfile --pmap-
   src-NETFLIX=AS2906,AS40027 --pass=passinnetflix.rw --
   fail=stdout   input  -pipe=stdin | rwfilter --type=in,
   inweb --site-config-file=$SILKCONF --pmap-file=AKAMAI:
   $pmapfile --pmap-src-AKAMAI=AS12222,AS16625,AS18680,
   AS18717,AS20189,AS20940,AS24319,AS31108,AS31109,
   AS31110,AS31377,AS32787,AS33905,AS34164,AS35204,
   AS35993,AS35994,AS43639,AS49846,AS21342,AS22452,
   AS23454,AS26008 --pass=passinakamai.rw --fail=stdout
   --input-pipe=stdin | rwfilter --type=in,inweb --site-
   config-file=$SILKCONF --pmap-file=GOOGLEY:$pmapfile --
   pmap-src-GOOGLEY=AS15169,AS19527,AS24424,AS36040,
   AS36384,AS36385,AS36492,AS41264,AS43515 --pass=
   passingoogley.rw --fail=stdout --input-pipe=stdin |
   rwfilter --type=in,inweb --site-config-file=$SILKCONF
   --pmap-file=APPLE:$pmapfile --pmap-src-APPLE=AS6185,
   AS714 --pass=passinapple.rw --fail=stdout --input-pipe
   =stdin | rwfilter --type=in,inweb --site-config-file=
   $SILKCONF --pmap-file=EDGECASTV:$pmapfile --pmap-src-
   EDGECASTV=AS15133 --pass=passinedgecastv.rw --fail=
   stdout --input-pipe=stdin | rwfilter --type=in,inweb
   --site-config-file=$SILKCONF --pmap-file=MSFT:
   $pmapfile --pmap-src-MSFT=AS12076,AS23468,AS35106,
   AS3598,AS45139,AS5761,AS58862,AS59067,AS6182,AS6194,
   AS6291,AS63314,AS6584,AS8068,AS8069,AS8070,AS8071,
   AS8075,AS8812 --pass=passinmsft.rw --fail=stdout --
   input-pipe=stdin | rwfilter --type=in,inweb --site-
   config-file=$SILKCONF --pmap-file=FASTLY:$pmapfile --
   pmap-src-FASTLY=AS54113 --pass=passinfastly.rw --fail=
   stdout --input-pipe=stdin | rwfilter --type=in,inweb
   --site-config-file=$SILKCONF --pmap-file=STACKPATH:
   $pmapfile --pmap-src-STACKPATH=AS11588,AS12989,AS18607
   ,AS20446,AS33438,AS64259 --pass=passinstackpath.rw --
   fail=stdout --input-pipe=stdin | rwfilter --type=in,
   inweb --site-config-file=$SILKCONF --pmap-file=
   REFLECTED:$pmapfile --pmap-src-REFLECTED=AS29789 --
   pass=passinreflected.rw --fail=stdout --input-pipe=
   stdin | rwfilter --type=in,inweb --site-config-file=
   $SILKCONF --pmap-file=LIMELIGHT:$pmapfile --pmap-src-
   LIMELIGHT=AS22822,AS23059,AS25804,AS38621,AS38622,
   AS45396,AS55429,AS60261 --pass=passinlimelight.rw --
   fail=stdout --input-pipe=stdin | rwfilter --type=in,
   inweb --site-config-file=$SILKCONF --pmap-file=
   CLOUDFLARE:$pmapfile --pmap-src-CLOUDFLARE=AS13335 --
   pass=passincloudflare.rw --fail=stdout --input-pipe=
   stdin | rwfilter --type=in,inweb --site-config-file=
   $SILKCONF --pmap-file=AMAZON:$pmapfile --pmap-src-
   AMAZON=AS16509,AS14618,AS55960,AS8987,AS19047,AS7224
   --pass=passinamazon.rw --fail=stdout --input-pipe=
   stdin | rwfilter --type=in,inweb --site-config-file=
   $SILKCONF --pmap-file=LEVEL:$pmapfile --pmap-src-LEVEL
   =AS3356,AS3549,AS12576 --pass=passinlevel.rw --fail=
   stdout --input-pipe=stdin | rwfilter --type=in,inweb
   --site-config-file=$SILKCONF --pmap-file=DROPBOX:
   $pmapfile --pmap-src-DROPBOX=AS19679,AS54372,AS62190
```

```
    --pass=passindropbox.rw --fail=stdout --input-pipe=
    stdin | rwfilter --type=in,inweb --site-config-file=
    $SILKCONF --pmap-file=TWITCH:$pmapfile --pmap-src-
    TWITCH=AS46849 --pass=passintwitch.rw --fail=stdout --
    input-pipe=stdin | rwfilter --type=in,inweb --site-
    config-file=$SILKCONF --pmap-file=ROKLA:$pmapfile --
    pmap-src-ROKLA=AS54994,AS60068,AS24429,AS23393,AS30081
    ,AS34789,AS10310,AS14776,AS14777,AS14778,AS14779,
    AS14780,AS14781,AS23879,AS24506,AS26085,AS26101,
    AS34010,AS36088,AS36229,AS36646,AS36647,AS56173,AS5779
    ,AS7233,AS40009,AS59019,AS21245,AS57363,AS60626,
    AS21030,AS19551,AS62571,AS33047,AS11855,AS12179,
    AS12180,AS12182,AS13789,AS13792,AS14742,AS14744,
    AS14745,AS15570,AS17675,AS19024,AS24245,AS24246,
    AS24247,AS30282,AS48910,AS36408,AS38107,AS63541,
    AS52580,AS22518,AS40428 --pass=passinrokla.rw --fail=
    failinrokla.rw --input-pipe=stdin
rwfilter failinrokla.rw --type=in,inweb --site-config-
    file=$SILKCONF --pmap-file=TELENOR:$pmapfile --pmap-
    src-TELENOR=AS2119 --pass=passintelenor.rw
echo rwfilter done!
rwuniq passinfb.rw --pmap-file=FACEBOOK:$pmapfile --
    fields=src-FACEBOOK --values=records,bytes,packets --
    no-col --output=passinfb.txt --copy-input=stdout |
    rwcount --bin-size=$BIN --no-col --column-separator
    =";" > countinfb.txt
echo Facebook done!
rwuniq passinnetflix.rw --pmap-file=NETFLIX:$pmapfile --
    fields=src-NETFLIX --values=records,bytes,packets --no
    -col --output=passinnetflix.txt --copy-input=stdout |
    rwcount --bin-size=$BIN --no-col --column-separator
    =";" > countinnetflix.txt
echo Netflix done!
rwuniq passinakamai.rw --pmap-file=AKAMAI:$pmapfile --
    fields=src-AKAMAI --values=records,bytes,packets --no-
    col --output=passinakamai.txt --copy-input=stdout |
    rwcount --bin-size=$BIN --no-col --column-separator
    =";" > countinakamai.txt
echo Akamai done!
#AS17204 removed
```

```
rwuniq passingoogley.rw --pmap-file=GOOGLEY:$pmapfile --
   fields=src-GOOGLEY --values=records,bytes,packets --no
   -col --output=passingoogley.txt --copy-input=stdout |
   rwcount --bin-size=$BIN --no-col --column-separator
   =";" > countingoogley.txt
echo Google Youtube done!
rwuniq passinapple.rw --pmap-file=APPLE:$pmapfile --
   fields=src-APPLE --values=records,bytes,packets --no-
   col --output=passinapple.txt --copy-input=stdout |
   rwcount  --bin-size=$BIN --no-col --column-separator
   =";" > countinapple.txt
echo Apple done!
rwuniq passinedgecastv.rw --pmap-file=EDGECASTV:$pmapfile
    --fields=src-EDGECASTV --values=records,bytes,packets
    --no-col --output=passinedgecastv.txt --copy-input=
   stdout | rwcount --bin-size=$BIN --no-col --column-
   separator=";" > countinedgecastv.txt
echo Edgecast Verizon done!
#AS13399 removed
rwuniq passinmsft.rw --pmap-file=MSFT:$pmapfile --fields=
   src-MSFT --values=records,bytes,packets --no-col --
   output=passinmsft.txt --copy-input=stdout | rwcount --
   bin-size=$BIN --no-col --column-separator=";" >
   countinmsft.txt
echo Microsoft Azure done!
rwuniq passinfastly.rw --pmap-file=FASTLY:$pmapfile --
   fields=src-FASTLY --values=records,bytes,packets --no-
   col --output=passinfastly.txt --copy-input=stdout |
   rwcount --bin-size=$BIN --no-col --column-separator
   =";" > countinfastly.txt
echo Fastly done!
rwuniq passinstackpath.rw --pmap-file=STACKPATH:$pmapfile
    --fields=src-STACKPATH --values=records,bytes,packets
    --no-col --output=passinstackpath.txt --copy-input=
   stdout | rwcount --bin-size=$BIN --no-col --column-
   separator=";" > countinstackpath.txt
echo Stackpath done!
rwuniq passinreflected.rw --pmap-file=REFLECTED:$pmapfile
    --fields=src-REFLECTED --values=records,bytes,packets
    --no-col --output=passinreflected.txt --copy-input=
   stdout | rwcount --bin-size=$BIN --no-col --column-
   separator=";" > countinreflected.txt
echo Reflected Networks done!
#AS37277 removed
rwuniq passinlimelight.rw --pmap-file=LIMELIGHT:$pmapfile
     --fields=src-LIMELIGHT --values=records,bytes,packets
     --no-col --output=passinlimelight.txt --copy-input=
   stdout | rwcount --bin-size=$BIN --no-col --column-
   separator=";" > countinlimelight.txt
echo Limelight done!
rwuniq passincloudflare.rw --pmap-file=CLOUDFLARE:
   $pmapfile --fields=src-CLOUDFLARE --values=records,
   bytes,packets --no-col --output=passincloudflare.txt
   --copy-input=stdout | rwcount --bin-size=$BIN --no-col
     --column-separator=";" > countincloudflare.txt
echo Cloudflare done!
#AS9059 removed
rwuniq passinamazon.rw --pmap-file=AMAZON:$pmapfile --
   fields=src-AMAZON --values=records,bytes,packets --no-
   col --output=passinamazon.txt --copy-input=stdout |
   rwcount --bin-size=$BIN --no-col --column-separator
   =";" > countinamazon.txt
echo Amazon + AWS done!
#AS560 removed
rwuniq passinlevel.rw --pmap-file=LEVEL:$pmapfile --
   fields=src-LEVEL --values=records,bytes,packets --no-
   col --output=passinlevel.txt --copy-input=stdout |
   rwcount --bin-size=$BIN --no-col --column-separator
   =";" > countinlevel.txt
echo Level3 done!
rwuniq passindropbox.rw --pmap-file=DROPBOX:$pmapfile --
   fields=src-DROPBOX --values=records,bytes,packets --no
   -col --output=passindropbox.txt --copy-input=stdout |
   rwcount --bin-size=$BIN --no-col --column-separator
   =";" > countindropbox.txt
echo Dropbox done!
rwuniq passintwitch.rw --pmap-file=TWITCH:$pmapfile --
   fields=src-TWITCH --values=records,bytes,packets --no-
   col --output=passintwitch.txt --copy-input=stdout |
   rwcount --bin-size=$BIN --no-col --column-separator
   =";" > countintwitch.txt
echo Twitch done!
```

```
rwuniq passinrokla.rw --pmap-file=ROKLA:$pmapfile --
   fields=src-ROKLA --values=records,bytes,packets --no-
   col --output=passinrokla.txt --copy-input=stdout |
   rwcount --bin-size=$BIN --no-col --column-separator
   =";" > countinrokla.txt
echo Rokla done!
rwcount failinrokla.rw --bin-size=$BIN --no-col --column-
   separator=";" > countinfail.txt
echo Fail done!
rwuniq passintelenor.rw -pmap-file=TELENOR:$pmapfile --
   fields=src-TELENOR --values=records,bytes,packets --no
   -col --output=passintelenor.txt --copy-input=stdout |
   rwcount--bin-size=$BIN --no-col --column-separator=";"
    > countintelenor.txt
echo Telenor done!
echo IN DONE!
```

Appendix C
Count traffic

The file ”allpassfilesin.txt” is one of the arguments for the script ”rwcountallcn.sh” which is counting CDN traffic and not CDN traffic.

allpassfilesin.txt*

```
passinfb.rw
passinnetflix.rw
passinakamai.rw
passingoogley.rw
passinapple.rw
passindropbox.rw
passintwitch.rw
passinedgecastv.rw
passinmsft.rw
passinfastly.rw
passinstackpath.rw
passinreflected.rw
passinlimelight.rw
passincloudflare.rw
passinamazon.rw
passinlevel.rw
passinrokla.rw
```

*The file is corrected for the error mentioned in Section 4.3, where Twitch and Dropbox were written twice.

rwcountallcdn.sh

```
BIN=120
rwcount --xargs=allpassfilesext.txt --bin-size=$BIN --no-
   col --column-separator=";" > countallcdnext.txt
rwcount --xargs=allpassfilesin.txt --bin-size=$BIN --no-
   col --column-separator=";" > countpassallcdnin.txt
rwcount --xargs=allpassfilesout.txt --bin-size=$BIN --no-
   col --column-separator=";" > countpassallcdnout.txt
rwcount --xargs=allpassfiles.txt --bin-size=$BIN --no-col
    --column-separator=";" > countpassallcdnall.txt
rwcount --xargs=allfailfiles.txt --bin-size=$BIN --no-col
    --column-separator=";" > countfailcdnall.txt
```

Appendix D

Script for re-filtering Google, Netflix and Akamai

```
#rwfiltercdncache.sh
DATA= <PATH TO DATA-SET>
SILKCONF= <PATH TO SILK.CONF>
START=2020/03/11
END=2020/03/11
pmapfile=20200311.bgp.ripe.pmap
BIN=120
#Adapt ASN to the pmap-file used
echo Tests for Google Global Cache in NORDUnet, and other
     traffic from Google
rwfilter --data=$DATA --start=$START --end=$END --any-
    cidr=109.105.109.192/26,109.105.98.192/27 --type=
    ext2ext --pass=passextggc.rw --fail=stdout | rwfilter
    --type=ext2ext --site-config-file=$SILKCONF --pmap-
    file=GOOGLEY:$pmapfile --pmap-src-GOOGLEY=AS15169,
    AS19527,AS24424,AS36040,AS36384,AS36385,AS36492,
    AS41264,AS43515 --pass=passextsrcgoogleny.rw --fail=
    stdout --input-pipe=stdin | rwfilter --type=ext2ext --
    site-config-file=$SILKCONF --pmap-file=GOOGLEY:
    $pmapfile --pmap-dst-GOOGLEY=AS15169,AS19527,AS24424,
    AS36040,AS36384,AS36385,AS36492,AS41264,AS43515 --pass
    =passextdstgoogleny.rw --input-pipe=stdin
rwuniq passextsrcgoogleny.rw --pmap-file=GOOGLEY:
    $pmapfile --fields=src-GOOGLEY --values=records,bytes,
    packets --no-col --output=passextsrcgoogleny.txt --
    copy-input=stdout | rwcount --bin-size=$BIN --no-col
    --column-separator=";" > countextsrcgoogleny.txt
rwuniq passextdstgoogleny.rw --pmap-file=GOOGLEY:
    $pmapfile --fields=dst-GOOGLEY --values=records,bytes,
    packets --no-col --output=passextdstgoogleny.txt --
```

```
   copy-input=stdout | rwcount --bin-size=$BIN --no-col
   --column-separator=";" > countextdstgoogleny.txt
rwcount passextggc.rw --bin-size=$BIN --no-col --column-
   separator=";" > countextggc.txt
echo Google Youtube ext done!
rwfilter --data=$DATA --start=$START --end=$END --scidr
   =109.105.109.192/26,109.105.98.192/27 --type=in,inweb
   --pass=passinggc.rw --fail=stdout | rwfilter --type=in
   ,inweb --site-config-file=$SILKCONF --pmap-file=
   GOOGLEY:$pmapfile --pmap-src-GOOGLEY=AS15169,AS19527,
   AS24424,AS36040,AS36384,AS36385,AS36492,AS41264,
   AS43515 --pass=passingoogleny.rw --input-pipe=stdin
rwuniq passingoogleny.rw --pmap-file=GOOGLEY:$pmapfile --
   fields=src-GOOGLEY --values=records,bytes,packets --no
   -col --output=passingoogleny.txt --copy-input=stdout |
    rwcount --bin-size=$BIN --no-col --column-separator
   =";" > countingoogleny.txt
rwcount passinggc.rw --bin-size=$BIN --no-col --column-
   separator=";" > countinggc.txt
echo Google Youtube in done!
rwfilter --data=$DATA --start=$START --end=$END --dcidr
   =109.105.109.192/26,109.105.98.192/27 --type=out,
   outweb --pass=passoutggc.rw --fail=stdout | rwfilter
   --type=out,outweb --site-config-file=$SILKCONF --pmap-
   file=GOOGLEY:$pmapfile --pmap-dst-GOOGLEY=AS15169,
   AS19527,AS24424,AS36040,AS36384,AS36385,AS36492,
   AS41264,AS43515 --pass=passoutgoogleny.rw --input-pipe
   =stdin
rwuniq passoutgoogleny.rw --pmap-file=GOOGLEY:$pmapfile
   --fields=dst-GOOGLEY --values=records,bytes,packets --
   no-col --output=passoutgoogleny.txt --copy-input=
   stdout | rwcount --bin-size=$BIN --no-col --column-
   separator=";" > countoutgoogleny.txt
rwcount passoutggc.rw --bin-size=$BIN --no-col --column-
   separator=";" > countoutggc.txt
echo Google Youtube out done!
echo Tests for Netflix cache in NORDUnet, and other
   traffic from Netflix
rwfilter --data=$DATA --start=$START --end=$END --any-
   cidr=109.105.98.160/31,109.105.98.158/31 --type=
   ext2ext --pass=passextnetflixcache.rw --fail=stdout |
   rwfilter --type=ext2ext --site-config-file=$SILKCONF
   --pmap-file=NETFLIX:$pmapfile --pmap-src-NETFLIX=
   AS2906,AS40027 --pass=passextsrcnetflixny.rw --fail=
   stdout - input -pipe=stdin | rwfilter --type=ext2ext
   --site-config-file=$SILKCONF --pmap-file=NETFLIX:
   $pmapfile --pmap-dst-NETFLIX=AS2906,AS40027 --pass=
   passextdstnetflixny.rw - input -pipe=stdin
rwuniq passextsrcnetflixny.rw --pmap-file=NETFLIX:
   $pmapfile --fields=src-NETFLIX --values=records,bytes,
   packets --no-col --output=passextsrcnetflixny.txt --
   copy-input=stdout | rwcount --bin-size=$BIN --no-col
   --column-separator=";" > countextsrcnetflixny.txt
rwuniq passextdstnetflixny.rw --pmap-file=NETFLIX:
   $pmapfile --fields=dst-NETFLIX --values=records,bytes,
   packets --no-col --output=passextdstnetflixny.txt --
   copy-input=stdout | rwcount --bin-size=$BIN --no-col
   --column-separator=";" > countextdstnetflixny.txt
rwcount passextnetflixcache.rw --bin-size=$BIN --no-col
   --column-separator=";" > countextnetflixcache.txt
echo Netflix ext done!
rwfilter --data=$DATA --start=$START --end=$END --scidr
   =109.105.98.160/31,109.105.98.158/31 --type=in,inweb
   --pass=passinnetflixcache.rw --fail=stdout | rwfilter
   --type=in,inweb --site-config-file=$SILKCONF --pmap-
   file=NETFLIX:$pmapfile --pmap-src-NETFLIX=AS2906,
   AS40027 --pass=passinnetflixny.rw   -input-pipe=stdin
rwuniq passinnetflixny.rw --pmap-file=NETFLIX:$pmapfile
   --fields=src-NETFLIX --values=records,bytes,packets --
   no-col --output=passinnetflixny.txt --copy-input=
   stdout | rwcount --bin-size=$BIN --no-col --column-
   separator=";" > countinnetflixny.txt
rwcount passinnetflixcache.rw --bin-size=$BIN --no-col --
   column-separator=";" > countinnetflixcache.txt
echo Netflix in done!
rwfilter --data=$DATA --start=$START --end=$END --dcidr
   =109.105.98.160/31,109.105.98.158/31 --type=out,outweb
    --pass=passoutnetflixcache.rw --fail=stdout |
   rwfilter --type=out,outweb --site-config-file=
   $SILKCONF --pmap-file=NETFLIX:$pmapfile --pmap-dst-
   NETFLIX=AS2906,AS40027 --pass=passoutnetflixny.rw --
   input-pipe=stdin
```

```
rwuniq passoutnetflixny.rw --pmap-file=NETFLIX:$pmapfile
   --fields=dst-NETFLIX --values=records,bytes,packets --
   no-col --output=passoutnetflixny.txt --copy-input=
   stdout | rwcount --bin-size=$BIN --no-col --column-
   separator=";" > countoutnetflixny.txt
rwcount passoutnetflixcache.rw --bin-size=$BIN --no-col
   --column-separator=";" > countoutnetflixcache.txt
echo Netflix out done!
echo Tests for Akamai cache in NORDUnet, and other
   traffic from Akamai
rwfilter --data=$DATA --start=$START --end=$END --scidr
   =109.105.109.0/26 --type=in,inweb --pass=
   passinakamaicache.rw --fail=stdout | rwfilter --type=
   in,inweb --site-config-file=$SILKCONF --pmap-file=
   AKAMAI:$pmapfile --pmap-src-AKAMAI=AS12222,AS16625,
   AS18680,AS18717,AS20189,AS20940,AS24319,AS31108,
   AS31109,AS31110,AS31377,AS32787,AS33905,AS34164,
   AS35204,AS35993,AS35994,AS43639,AS49846,AS21342,
   AS22452,AS23454,AS26008 --pass=passinakamainy.rw --
   input-pipe=stdin
rwuniq passinakamainy.rw --pmap-file=AKAMAI:$pmapfile --
   fields=src-AKAMAI --values=records,bytes,packets --no-
   col --output=passinakamainy.txt --copy-input=stdout |
   rwcount --bin-size=$BIN --no-col --column-separator
   =";" > countinakamainy.txt
rwcount passinakamaicache.rw --bin-size=$BIN --no-col --
   column-separator=";" > countinakamaicache.txt
echo Akamai in done!
rwfilter --data=$DATA --start=$START --end=$END --dcidr
   =109.105.109.0/26 --type=out,outweb --pass=
   passoutakamaicache.rw --fail=stdout | rwfilter --type=
   out,outweb --site-config-file=$SILKCONF --pmap-file=
   AKAMAI:$pmapfile --pmap-dst-AKAMAI=AS12222,AS16625,
   AS18680,AS18717,AS20189,AS20940,AS24319,AS31108,
   AS31109,AS31110,AS31377,AS32787,AS33905,AS34164,
   AS35204,AS35993,AS35994,AS43639,AS49846,AS21342,
   AS22452,AS23454,AS26008 --pass=passoutakamainy.rw --
   input-pipe=stdin
rwuniq passoutakamainy.rw --pmap-file=AKAMAI:$pmapfile --
   fields=dst-AKAMAI --values=records,bytes,packets --no-
   col --output=passoutakamainy.txt --copy-input=stdout |
    rwcount --bin-size=$BIN --no-col --column-separator
   =";" > countoutakamainy.txt
rwcount passoutakamaicache.rw --bin-size=$BIN --no-col --
   column-separator=";" > countoutakamaicache.txt
echo Akamai out done!
rwfilter --data=$DATA --start=$START --end=$END --any-
   cidr=109.105.109.0/26 --type=ext2ext --pass=
   passextakamaicache.rw --fail=stdout | rwfilter --type=
   ext2ext --site-config-file=$SILKCONF --pmap-file=
   AKAMAI:$pmapfile --pmap-src-AKAMAI=AS12222,AS16625,
   AS18680,AS18717,AS20189,AS20940,AS24319,AS31108,
   AS31109,AS31110,AS31377,AS32787,AS33905,AS34164,
   AS35204,AS35993,AS35994,AS43639,AS49846,AS21342,
   AS22452,AS23454,AS26008 --pass=passextsrcakamainy.rw
   --fail=stdout --input-pipe=stdin | rwfilter --type=
   ext2ext --site-config-file=$SILKCONF --pmap-file=
   AKAMAI:$pmapfile --pmap-dst-AKAMAI=AS12222,AS16625,
   AS18680,AS18717,AS20189,AS20940,AS24319,AS31108,
   AS31109,AS31110,AS31377,AS32787,AS33905,AS34164,
   AS35204,AS35993,AS35994,AS43639,AS49846,AS21342,
   AS22452,AS23454,AS26008 --pass=passextdstakamainy.rw
   --input-pipe=stdin
rwuniq passextsrcakamainy.rw --pmap-file=AKAMAI:$pmapfile
    --fields=src-AKAMAI --values=records,bytes,packets --
   no-col --output=passextsrcakamainy.txt --copy-input=
   stdout | rwcount --bin-size=$BIN --no-col --column-
   separator=";" > countextsrcakamainy.txt
rwuniq passextdstakamainy.rw --pmap-file=AKAMAI:$pmapfile
    --fields=dst-AKAMAI --values=records,bytes,packets --
   no-col --output=passextdstakamainy.txt --copy-input=
   stdout | rwcount --bin-size=$BIN --no-col --column-
   separator=";" > countextdstakamainy.txt
rwcount passextakamaicache.rw --bin-size=$BIN --no-col --
   column-separator=";" > countextakamaicache.txt
echo Akamai ext done!
echo ALL DONE!
```

Appendix E
Script for re-filtering Twitch

```
#rwfiltertwitch.sh
DATA= <PATH TO DATA-SET>
SILKCONF= <PATH TO SILK.CONF>
START=2020/03/11
END=2020/03/11
pmapfile=20200311.bgp.ripe.pmap
BIN=120
echo Retest Twitch, and counting all traffic:
rwfilter --data=$DATA --start=$START --end=$END --type=in
   ,inweb,out,outweb,int2int,ext2ext --pass=
   countalltraffic.rw
rwcount countalltraffic.rw --bin-size=$BIN --no-col --
   column-separator=";" > countalltraffic.txt
echo Count all traffic done!
rwfilter --type=in,inweb ---start=$START --end=$END --
   pmap-file=TWITCH:$pmapfile --pmap-src-TWITCH=AS46489
   --pass=passintwitch.rw --data=$DATA
rwuniq passintwitch.rw --pmap-file=TWITCH:$pmapfile --
   fields=src-TWITCH --values=records,bytes,packets --no-
   col --output=passintwitch.txt --copy-input=stdout |
   rwcount --bin-size=$BIN --no-col --column-separator
   =";" > countintwitch.txt
echo Twitch in done!
rwfilter --type=out,outweb --start=$START --end=$END --
   pmap-file=TWITCH:$pmapfile --pmap-dst-TWITCH=AS46489
   --pass=passouttwitch.rw --data=$DATA
rwuniq passouttwitch.rw --pmap-file=TWITCH:$pmapfile --
   fields=dst-TWITCH --values=records,bytes,packets --no-
   col --output=passouttwitch.txt --copy-input=stdout |
   rwcount --bin-size=$BIN --no-col --column-separator
   =";" > countouttwitch.txt
echo Twitch out done!
rwfilter --type=ext2ext --start=$START --end=$END --pmap-
   file=TWITCH:$pmapfile --pmap-dst-TWITCH=AS46489 --pass
   =passextdsttwitch.rw --data=$DATA
rwuniq passextdsttwitch.rw --pmap-file=TWITCH:$pmapfile
   --fields=dst-TWITCH --values=records,bytes,packets --
   no-col --output=passextdsttwitch.txt --copy-input=
   stdout | rwcount --bin-size=$BIN --no-col --column-
   separator=";" > countextdsttwitch.txt
echo ext dst Twitch done!
rwfilter --type=ext2ext --start=$START --end=$END --pmap-
   file=TWITCH:$pmapfile --pmap-src-TWITCH=AS46489 --pass
   =passextsrctwitch.rw --data=$DATA
rwuniq passextsrctwitch.rw --pmap-file=TWITCH:$pmapfile
   --fields=src-TWITCH --values=records,bytes,packets --
   no-col --output=passextsrctwitch.txt --copy-input=
   stdout | rwcount --bin-size=$BIN --no-col --column-
   separator=";" > countextsrctwitch.txt
echo ext src Twitch done!
```

Sissel-Johanne Aspdal

Optimized use of CDN technologies in Uninett

NTNU
Norwegian University of
Science and Technology