EXAMENSARBETE INOM ELEKTROTEKNIK,
GRUNDNIVÅ, 15 HP
STOCKHOLM, *SVERIGE 2016*

Distribuerade datalagringssystem för tjänsteleverantörer

Distributed Data Storage Systems for Service Providers

Undersökning av olika användningsfall för distribuerade datalagringssystem

Investigation of different use cases for distributed data storage systems

BRATISLAV MARKOVIC

TANVIR SAIF AHMED

KTH
SKOLAN FÖR TEKNIK OCH HÄLSA

Distribuerade datalagringssystem för tjänsteleverantörer
Distributed Data Storage Systems for Service Providers

Undersökning av olika användningsfall för distribuerade datalagringssystem

Investigation of different use cases for distributed data storage systems

Tanvir Saif Ahmed
Bratislav Markovic

Examensarbete inom elektroteknik, grundnivå
HE110X, 15,0 HP
Handledare på KTH: Ibrahim Orhan
Examinator: Thomas Lindh
TRITA-STH 2016:110

KTH
Skolan för Teknik och Hälsa
136 40 Handen, Sverige

Sammanfattning

Detta examensarbete handlar om undersökning av tre olika användningsfall inom datalagring; *Cold Storage, High Performance Storage* och *Virtual Machine Storage*. Rapporten har som syfte att ge en översikt över kommersiella distribuerade filsystem samt en djupare undersökning av distribuerade filsystem som bygger på öppen källkod och därmed hitta en optimal lösning för dessa användningsfall. I undersökningen ingick att analysera och jämföra tidigare arbeten där jämförelser mellan prestandamätningar, dataskydd och kostnader utfördes samt lyfta upp diverse funktionaliteter (snapshotting, multi-tenancy, datadeduplicering, datareplikering) som moderna distribuerade filsystem kännetecknas av. Både kommersiella och öppna distribuerade filsystem undersöktes. Även en kostnadsuppskattning för kommersiella och öppna distribuerade filsystem gjordes för att ta reda på lönsamheten för dessa två typer av distribuerat filsystem.

Efter att jämförelse och analys av olika tidigare arbeten utfördes, visade sig att det öppna distribuerade filsystemet Ceph lämpade sig bra som en lösning utifrån kraven som sattes som mål för *High Performance Storage* och *Virtual Machine Storage*. Kostnadsuppskattningen visade att det var mer lönsamt att implementera ett öppet distribuerat filsystem.

Denna undersökning kan användas som en vägledning vid val mellan olika distribuerade filsystem.

Nyckelord

Cold Storage, High Performance Storage, Virtual Machine Storage, användningsfall, snapshotting, multi-tenancy, datadeduplicering, datareplikering, distribuerade filsystem.

Abstract

In this thesis, a study of three different uses cases has been made within the field of data storage, which are as following: *Cold Storage*, *High Performance Storage* and *Virtual Machine Storage*. The purpose of the survey is to give an overview of commercial distributed file systems and a deeper study of open source codes distributed file systems in order to find the most optimal solution for these use cases. Within the study, previous works concerning performance, data protection and costs were analyzed and compared in means to find different functionalities (snapshotting, multi-tenancy, data duplication and data replication) which distinguish modern distributed file systems. Both commercial and open distributed file systems were examined. A cost estimation for commercial and open distributed file systems were made in means to find out the profitability for these two types of distributed file systems.

After comparing and analyzing previous works, it was clear that the open source distributed file system Ceph was proper as a solution in accordance to the objectives that were set for *High Performance Storage* and *Virtual Machine Storage*. The cost estimation showed that it was more profitable to implement an open distributed file system.

This study can be used as guidance to choose between different distributed file systems.

Keywords

Cold Storage, High Performance Storage, Virtual Machine Storage, uses cases, snapshotting, multi-tenancy, data deduplication, data replication, distributed file systems.

Förord

Härmed vill vi tacka våra handledare Ibrahim Orhan från KTH STH och Fredrik Hellgren från AxByte AB för enormt stöd under examensarbetet.

Samling över förkortningar

Nedanstående tabell visar de förkortningar på termer som tas upp i examensarbetet.

Förkortning	Term	Definition
IOPS	Input/Output Operations Per Second	Antalet skrivningar/läsningar på en disk för ett datalagringssystem.
QoS	Quality of Service	En övergripande prestanda av ett system.
SAN	Storage Area Network	Höghastighetsdatornätverk som ger tillgång till blockbaserad lagring.
FC	Fiber Channel	Nätverksprotokoll för överföringshastigheter i storleksordningen 1-16 Gb/s.
AoE	ATA over Ethernet	Nätverksprotokoll skapad för att koppla samman olika lagringsenheter i SAN.
iSCSI	Internet Small Computer Systems Interface	Transportlagerprotokoll som beskriver hur SCSI paket bör transporteras över TCP/IP.
CIFS	Common Internet File System	Standardprotokoll för delning av filer över nätet.
NFS	Network File System	Protokoll som används för delning av mappar och filer över nätet.
SMB	Server Message Block	Protokoll som används för att dela dataresurser, exempelvis filer eller skrivare mellan olika enheter i ett datornätverk.
NAS	Network Attached Storage	Datorenhet som fungerar som en central lagringsenhet uppkopplad till ett nätverk.

RAID	Redundant Array of Independent Disks	En teknik som representerar en array av diskar som en enhetlig disk till operativsystemet.
LUN	Logical unit number	En identifierare för att beteckna en samling av fysiska eller virtuella lagringsenheter.
SLA	Service Level Agreement	En överenskommelse mellan två eller fler parter där ena är kund och andra är tjänsteleverantör.
DFS	Distributed File System	Typ av distribuerat filsystem: Ceph, SwiftStack, GlusterFS.
POSIX	Portable Operating System Interface	Samling av standarder specifierade av IEEE för att definiera API.

Innehållsförteckning

1 Inledning

I detta kapitel presenteras problemformulering, målen och avgränsningar samt de bidrag som författarna har tillfört till examensarbetet.

1.1 Problemformulering

Tjänsteleverantörer som arbetar med datalagringssystem möter en alltmer krävande marknad idag, där bättre prestanda, lägre kostnader och högre dataskydd är en stor utmaning. Datahastigheter som erbjuds samt de mängder av data som växer alltmer leder till att de kraven som tjänsteleverantörer måste uppfylla blir allt högre. Mängden av data som en vanlig användare genererar från dag till dag genom att använda elektronikprylar som exempelvis mobiltelefoner, surfplattor, växer snabbt och dess data kommer att förr eller senare lagras i molnet för framtida användningar. Nya lagringstekniker som Big Data[1] kräver behandling av enorm data under ett kort tidsintervall men också möjligheten för vidare analys.

Svenska tjänsteleverantören AxByte vill undersöka olika användningsfall för datalagringssystem för att kunna utveckla optimala lösningar som passar ett specifikt datalagringssystem. Detta examensarbete ska ge en översikt av de olika datalagringstekniker som AxByte kan tillämpa för olika typer av datalagringstjänster.

1.2 Målsättning

Målet med examensarbetet är att undersöka olika användningsfall för distribuerade filsystem. I undersökningen kommer det att ingå analys och jämförelse av distribuerade filsystem som bygger på öppen källkod samt kommersiella distribuerade filsystem. En djupare undersökning kommer att utföras endast på öppna distribuerade filsystem (se avgränsningar). Denna undersökning kommer att användas som en grund för att kunna fastställa ett optimalt öppet distribuerat filsystem som kan tillämpas för ett datalagringssystem beroende på användningsfall som tas i beaktande. De användningsfallen som kommer att behandlas i detta examensarbete för var och en av öppna distribuerade filsystemen är *Cold Storage*, *High Performance Storage* samt *Virtual Machine Storage*. Valda användningsfall täcker växande behov som dagens användare har beroende på vilken typ av lagring som efterfrågas. Exempelvis lämpas *Cold Storage* bäst för lagring av infrekvent data, där låg kostnad och skydd av data är högst prioritet. Å andra sidan kännetecknas *High Performance Storage* av systemprestanda som högst prioritet, där systemet förväntas svara snabbt. Denna egenskap innebär högre kostnad. Gällande *Virtual Machine Storage* är prestandaisolering en viktig aspekt, där systemet förväntas stödja hundratals användare utan att prestandan försämras för var och en av dem. De parametrarna som kommer att utgås ifrån är prestanda, kostnad och dataskydd.

[1] **Big Data** är en term som används inom databehandling då det handlar om enorma och komplexa datamängder [1].

1.3 Avgränsningar

Då arbetsgivaren AxByte vill ta fram egna tjänster för de olika användningsfallen med hjälp av lösningar som tillhandahålls av öppna distribuerade filsystem, kommer examensarbetarna att lägga störst fokus på djupare analys och jämförelse av distribuerade filsystem som bygger på öppen källkod. Bland andra öppna distribuerade filsystem som kommer att undersökas är Ceph, SwiftStack och GlusterFS. Kommersiella distribuerade filsystem kommer enbart att analyseras och jämföras för att ge en överblick på de riktlinjer som examensarbetarna ska följa under undersökning av öppna distribuerade filsystem. Examensarbetet omfattar 10 veckors heltidsstudier.

1.4 Författarnas bidrag till examensarbetet

Examensarbetarna har jobbat självständigt och arbetet har varit jämnfördelade. Litteraturer, tidigare arbeten samt tillgänglig dokumentation har studerats. Olika lösningar som bygger på öppen källkod samt kommersiella lösningar har analyserats och jämförts.

2 Teori och bakgrund

I detta kapitel, presenteras en introduktion till distribuerad datalagring, beskrivning om tidigare arbeten kring datalagring samt en teknisk beskrivning av de olika användningsfallen som studerats.

2.1 Distribuerad datalagring och dess parametrar

Följande avsnitt ger en introduktion till distribuerad datalagring samt definierar parametrar (prestanda och dataskydd) inom distribuerad datalagring.

2.1.1 Distribuerad datalagring

Lagring av data har utvecklats genom åren för att tillgodose ökade behov både från företag och privatpersoner. Denna utveckling har kommit till en punkt där snabbare enheter och högre nätverkshastighet inte längre duger för att göra ett datalagringsystem som uppfyller olika krav gällande prestanda, dataskydd och kostnader. Traditionella datalagringssystem bestod oftast av en server och tillgängligt lagringsutrymme. Dessa servrar var kraftfulla (CPU med hög klockfrekvens samt stora mängder av RAM minne) men även extremt dyra. Dessutom var sådana system inte skapade för att hantera datamängder i storleksordning av peta- eller exabytenivå och därmed har dessa servrar blivit en flaskhals för sådana system. Genom att lägga till fler lagringsenheter ökas det totala lagringsutrymmet men samtidigt uppstår det en prestandasänkning då servern har mer data att hantera [107, 108].

Distribuerade datalagringssystem bestående av flertal servrar med varsitt lagringsutrymme utgör ett kluster av separata servrar som fungerar som en helhet. Sådana system består av flertal servrar som kan vara utspridda genom flera geografiskt avlägsna platser. Dessa servrar klassas oftast som commodity hardware (se avsnitt 2.6.1.3) och därmed kan den totala kostnaden sänkas. Ytterligare en anledning är att dessa system är flexibla, det vill säga det är möjligt att lägga till fler lagringsenheter utan att prestandan försämras då dessa enheter är ”synliga” för samtliga servrar. Detta är en viktig faktor då det gäller dataskydd (se avsnitt 2.1.4). Fler servrar innebär att en viss datamängd kan replikeras genom flera enheter, vilket ökar skydd av data ifall någon av enheterna skulle gå ned [107, 108].

2.1.2 Distribuerade filsystem

Ett distribuerat filsystem är ett klient-serverbaserat filsystem där data lagras på en eller flera servrar där varje server har ett eget filsystem för hantering av data. Denna data hämtas och behandlas som om den var lagrad på den lokala klientdatorn. Distribuerade filsystem gör det bekvämt att dela information och filer mellan användare i ett nätverk på ett säkert sätt. Servern tillåter klienter att dela filer och lagra data precis som om de lagras lokalt. Dessutom har servrarna full kontroll över data och ger säker åtkomst till klienterna. Till skillnad från vanliga filsystem, tillåter distribuerade filsystem tillgång till filer som kan delas mellan flera användare via ett datornätverk [109, 110].

2.1.3 Vad innebär prestanda inom datalagring?

Några viktiga prestandaparametrar som är väsentliga att observera för att kunna mäta den allmänna prestandan i ett datalagringssystem, nämns nedan [2, 3, 111]:

- IOPS – Antalet skrivningar/läsningar på en disk för ett datalagringssystem.
- QoS – Kan hänvisas till prestanda, tillgängligheten eller den generella erfarenheten av tjänsten som en användare har.

Denna rapport kommer att behandla ovanstående parametrar, då dessa utgör grundläggande egenskaper när det gäller prestanda i ett datalagringssystem.

2.1.4 Vad innebär dataskydd inom datalagring?

Skydd av data innebär en process av att säkra viktig information så att det inte förloras eller blir korrupt. Ett sätt att skydda data kan vara att säkerställa att systemet stödjer katastrofåterställning (disaster recovery), det vill säga att det finns stöd för säkerhetskopiering av data.

Det finns några tekniker på hur data kan skyddas, bland annat är replikering ett sätt att skydda data, vilket innebär att det skapas kopior av data så att denna data är tillgänglig från mer än en fysisk nod i ett lagringssystem [4, 5].

Denna rapport kommer att behandla dataskydd utifrån två parametrar; datareplikering och dataåtkomst (multi-tenancy) då dessa utgör grundläggande egenskaper när det gäller skydd av data i ett datalagringssystem.

2.2 Tidigare arbeten

Följande avsnitt handlar om tidigare arbeten vars innehåll behandlar prestanda och dataskydd inom datalagring.

2.2.1 Designing Reliable High-Performance Storage Systems For HPC Environments

I arbetet “Designing Reliable High-Performance Storage Systems For HPC Environments” behandlades parametern dataskydd. Arbetet handlade om hur ett högprestanda datalagringssystem kan byggas samt bevaras på ett pålitligt vis. Bland annat behandlades RAID som ett dataskyddsalternativ. Dock visade sig i arbetet att RAID inte är något pålitligt dataskyddsalternativ. RAID kan bidra till att hålla data tillgänglig ifall en lagringsenhet går ned, däremot kan RAID inte skydda data om olika mjukvaru- och drivrutinsfel har uppstått [3]. Vidare i arbetet presenterades olika praktiska tekniker som kan implementeras på ett datalagringssystem som i sin tur kan förbättra systemets dataskydd. Bland annat olika mjukvarulösningar som har inbyggda datareplikeringsalgoritmer [3].

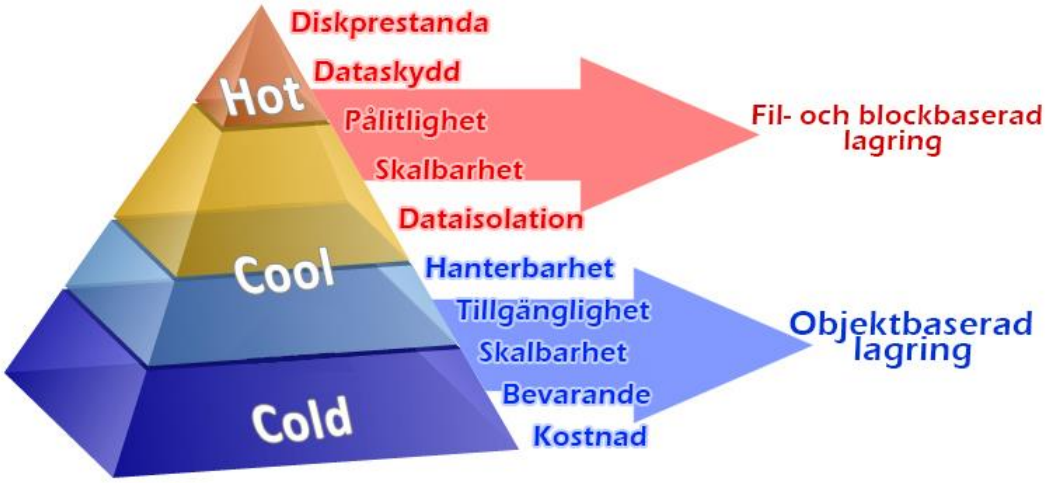
2.2.2 Vklass datalagring

I arbetet ”Vklass datalagring” behandlades samtliga parametrar prestanda, dataskydd och kostnad. Arbetet handlade om applikationen Vklass[2] där data lagrades lokalt och en säkerhetskopiering skedde regelbundet på en annan server. Det aktuella datalagringssystemet ansågs ineffektivt och oöverskådligt. De metoder som implementerades i arbetet var lokal datalagring och datalagring i molntjänster där jämförelse av olika lösningar av tre stora företag (Microsoft, Google och Amazon) utfördes. I analysen av de olika lösningsmetoderna, visade det sig att datalagring i molntjänster uppfyllde kraven som berörde prestanda och kostnad. Utifrån de lösningsmetoder som prestandamätningar och kostnadsjämförelser drogs en slutsats där Microsofts Azure Storage blev den mest lämpliga lösningen för datalagring då data lagrades via molntjänster och inte lokalt [6].

[2] **Vklass** är en applikation för skolor där lärare, elever och föräldrar samlas i en och samma portal för samarbete, kommunikation och dagligt skötande av administrativt arbete [6].

2.3 Olika nivåer av datalagring

Data kan lagras på olika nivåer i ett datalagringssystem. De nivåerna som finns inom datalagring är filbaserad lagring, blockbaserad lagring och objektbaserad lagring och dessa lagringsnivåer karakteriseras av några viktiga parametrar som visas i figur 1.

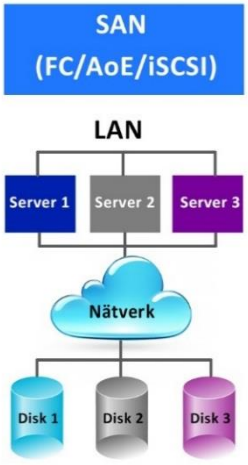


Figur 1: Figuren visar en illustration av de parametrarna som är typiska för olika nivåer av datalagring. Vid fil- och blockbaserad lagring är diskprestanda, dataskydd, pålitlighet de viktiga parametrarna. Kategorisering av lagringsnivåerna fil- och blockbaserad lagring kallas för ”Hot” och innebär att data används frekvent. Vid objektbaserad lagring är hög skalbarhet, tillgänglighet, lägre kostnad och databevaring de viktigaste parametrarna. Denna lagringsnivå kategoriseras som ”Cold” och innebär att data används väldigt sällan. Lagringsnivåerna fil-, block- och objektbaserad lagring kategoriseras som ”Cool” och är en blandning av kategorierna ”Hot” och ”Cold” [112].

2.3.1 Blockbaserad lagring

Blockbaserad lagring är en datalagringstyp där data lagras i olika block även kallade volymer [7, 8]. Denna typ av lagring är typisk för SAN miljöer, där system som implementerar denna teknik klassas som höghastighetssystem [8, 9]. Vidare beskrivs det att varje block uppträder som en egen hårddisk och konfigureras av en systemadministratör (se figur 2).

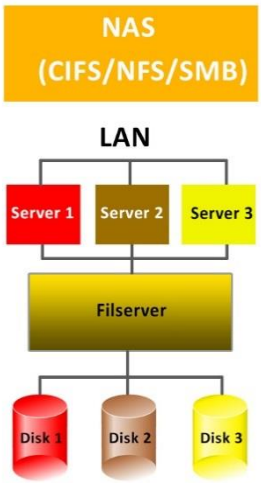
Dessa block kontrolleras av ett serverbaserat operativsystem och är oftast åtkomna antingen via FC-, AoE- eller iSCSI-protokollen. Blockbaserad lagring används ofta i system där låg latens och hög prestanda prioriteras [8, 10]. Förutom låg latens mellan servrar och lagringsmedium samt hög hastighet bör blockbaserade lagringssystem kännetecknas av garanterade IOPS, snapshots och datareplikeringsmetoder.



Figur 2: Figuren visar en illustration på hur blockbaserad lagring fungerar.

2.3.2 Filbaserad lagring

Filbaserad lagring innebär att data lagras i en hierarkisk struktur (se figur 3). Data som sparas i olika mappar och filer presenteras likadant för både systemet som data lagras i och systemet som hämtar denna data [8, 11]. För att komma åt data i ett lagringssystem som implementerar filbaserad lagring kan ett av tre följande protokollen användas: CIFS, NFS och SMB. NAS anses vara det bästa och därmed det säkraste sättet att tryggt dela filer mellan olika användare via nätet [12, 13].



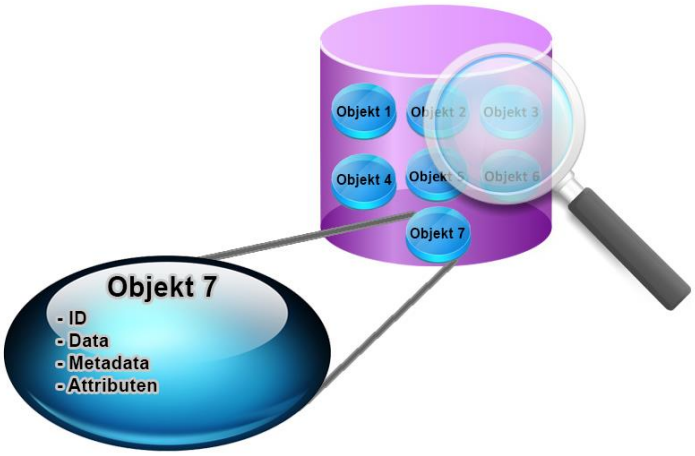
Figur 3: Figuren visar en illustration på hur filbaserad lagring fungerar.

Filbaserat lagringssystem fungerar bra då lagringssystemet hanterar tusentals och till och med miljontals mappar och filer, dock är sådana system inte utformade att hantera miljardtals filer. Dessa begränsningar kunde inte upptäckas förrän nyligen då företag som implementerade sådana system inte hade behov av system som kunde hantera antal mappar och filer av den storleksordningen.

2.3.3 Objektbaserad lagring

Objektbaserad lagring är ett enklare och mer skalbart lagringsalternativ än vanlig filbaserad lagring [14, 15]. Istället för att data ska organiseras som en hierarki av filkataloger som i den klassiska filhierarkin, organiseras data i en objektbaserad lagringsstruktur i form av objekt på en flat arkitektur. Flat arkitektur i sin tur innebär att varje objekt har ett unikt namn samt att ett objekt inte får innehålla ett annat objekt [15, 16]. Varje objekt har metadata ansluten till data som detta objekt innehåller (se figur 4).

Varje objekt har ett unikt ID som i sin tur möjliggör att servrar eller användare kan hämta data från olika objekt utan att behöva veta objektens fysiska adress [15, 17]. Till skillnad från filbaserad lagring, har objektbaserad lagring inte någon begränsning när det gäller antal ID det kan hantera. Detta gör att de system som implementerar objektbaserad lagring kan bli enormt skalbara utan att systemets prestanda försämras [15, 18].



Figur 4: Figuren visar en illustration på hur objektbaserad lagring fungerar.

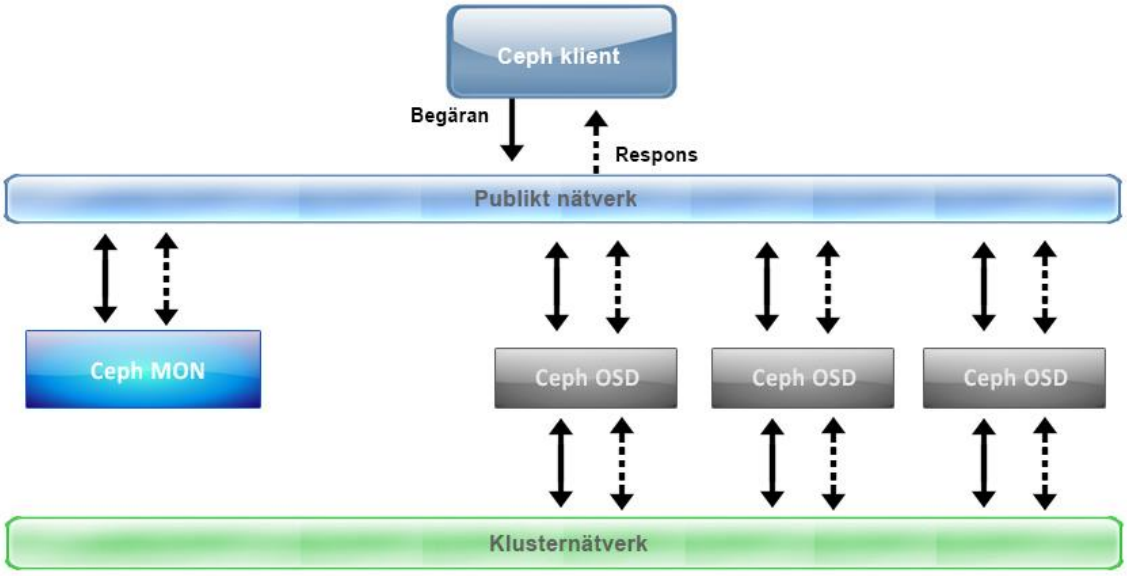
2.4 Öppna distribuerade filsystem

Då företag implementerar sina lösningar på olika sätt, har en beskrivning gjorts av följande öppna distribuerade filsystem som valts i detta examensarbete. Följande avsnitt ger en överblick på arkitekturen över valda öppna distribuerade filsystem.

2.4.1 Ceph

Ceph är ett distribuerat lagringssystem som bygger på öppen källkod skapat för högprestanda-, tillförlitlighet- och skalbarhetsystem. Detta system erbjuder block-, fil- och objektbaserad lagringssystem i en och samma plattform. Det är skapat att växa till exabytenivå och är tillräckligt feltolerant för att kunna köras på commodity hardware. RADOS (Reliable Autonomic Distributed Object Store) ligger på toppen av Ceph och hanterar alla tjänster som förses av Ceph [19]. De grundläggande byggstenarna i ett Cephsystem är (se figur 5):

- **OSD (Object Storage Daemons)** – lagrar data som objekt på lagringsnoder.
- **Monitors** – övervakar lagringssystemet och håller reda på systemets lediga utrymme.

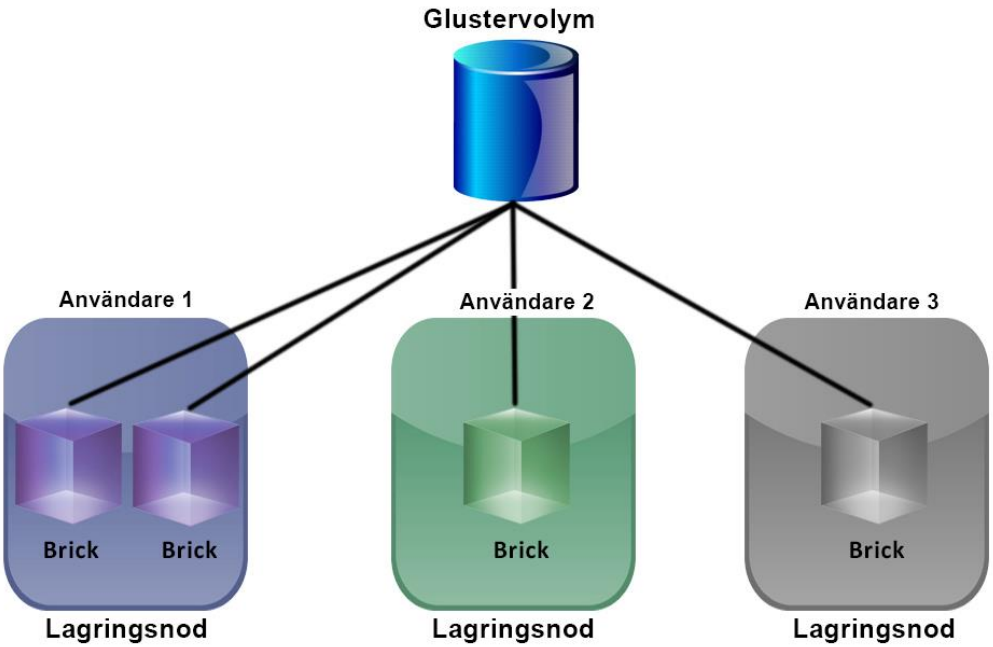


Figur 5: Figuren visar ett typiskt Cephsystem som består av ett antal OSDs och en monitor (Ceph MON) i klustret.

För att allokera data och enhetligt distribuera datakopior genom tillgängliga enheter för att upprätthålla balanserad lagringsutnyttjandegrad mellan enheterna, använder sig Ceph av CRUSH (Controlled Replication Under Scalable Hashing) algoritmen [20]. För Cephs arkitektur, se bilaga 1.

2.4.2 GlusterFS

GlusterFS är ett distribuerat lagringssystem som bygger på öppen källkod kapabel av att skalas upp till flera petabytenivå. System som implementerar GlusterFS kännetecknas av högprestanda-, skalbarhet- och datatillgänglighet. De grundläggande byggstenarna i ett GlusterFS system är bricks (se figur 6). Dessa motsvarar OSDs i Ceph och är en del av GlusterFS lagringsvolym.



Figur 6: Figuren visar hur ett typiskt GlusterFSsystem, bestående av fyra bricks ser ut.

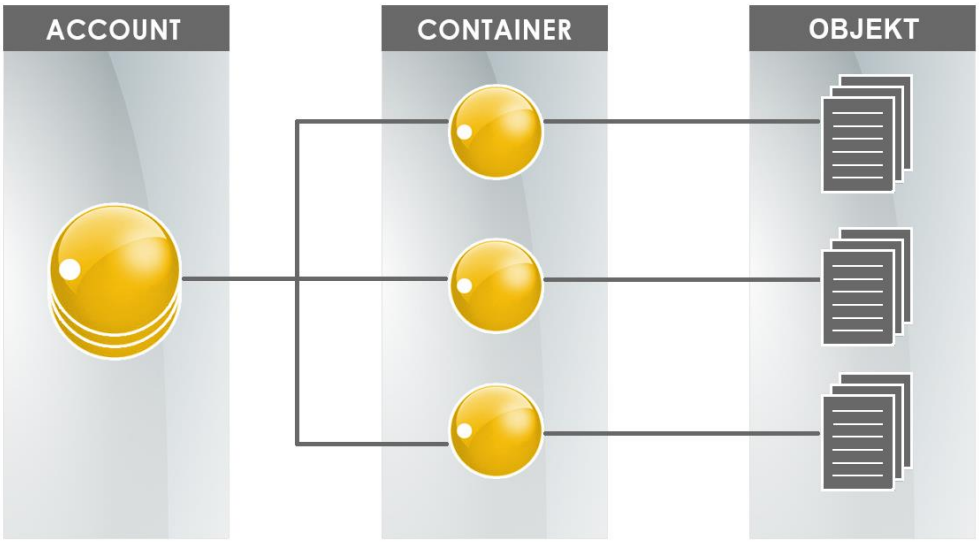
En annan viktig del av GlusterFS är EHA (Elastic Hash Algorithm), som är en algoritm för att allokera lagringsutrymme och enhetligt distribuera datakopior [21]. För GlusterFSs arkitektur, se bilaga 2.

2.4.3 SwiftStack

SwiftStack är ett distribuerat lagringssystem som bygger på öppen källkod och erbjuder enbart objektbaserad lösning. System som implementerar SwiftStack kan vara uppbyggda av några lagringsnoder upp till flera tusen lagringsnoder. Dessutom kan dessa noder vara geografisk utspridda och är kapabla att klara av exabytenivå data [22].

De grundläggande byggstenarna i ett SwiftStack system är (se figur 7):

- **Objekt** – representerar det verkliga data.
- **Container** – grupperar objekt.
- **Account** – grupperar containrar.



Figur 7: Figuren visar en illustration på hur data lagras i ett typiskt SwiftStacksystem. Data lagras i form av objekt och placeras i Containers, vars placering ligger i Accounts.

En annan viktig del av SwiftStack är Rings, som är en algoritm för att distribuera data i klustret. För SwiftStacks arkitektur, se bilaga 3.

2.5 Kommersiella distribuerade filsystem

Följande avsnitt ger en kort beskrivning av de datalagringslösningarna som tillhandahålls av de större företagen för att kunna ge en översikt av vad för typ av datalagringslösningar som erbjuds. Dessa är kommersiella lösningar och kräver oftast en licens för att lösningarna ska kunna implementeras i ett annat företag.

2.5.1 Google Cloud Storage

Google tillhandahåller datalagringstjänster till utvecklare samt IT-organisationer som är mycket hållbara och tillgängliga vid lagring av data. Tjänsterna kategoriseras vid namnet Google Cloud Storage och ger möjlighet till att lagra data samt komma åt data över hela världen i vilken tidpunkt som helst. Vidare så beskrivs det att Google ger möjligheten till att lagra data med en hög pålitlighet, prestanda samt tillgänglighet. Google Cloud Storage går att använda i olika typer av scenario, som omfattar t.ex. att uppehålla en hemsida, lagra data för arkivering och disaster recovery samt distribuera mängd med data till användare via direktnedladdning. Beroende på vad för typ av datalagrings-system som önskas, så erbjuder Google tre datalagringsslösningar, vilka är Standard Storage, Durable Reduced Availability (DRA) Storage och Nearline Storage [23].

- **Google Standard Storage**
  Denna datalagringstjänst kan tillämpas i de fallen där låglatens samt frekvent hämtning av data är ett krav. Exempel på fall där Standard Storage kan användas är t.ex. vid uppehåll av hemsidor samt mobila- och spelapplikationer [24].

- **Google Durable Reduced Availability (DRA) Storage**
  Denna datalagringstjänst tillämpas mest i de fallen där hög tillgänglighet och hög prestanda inte prioriteras, vilket innebär att tjänsten är av lågkostnad. I de situationer där DRA Storage kan användas är lagring av data som är kostnadskänsligt eller där hög tillgänglighet och hög prestanda inte eftersträvas [24].

- **Google Nearline Storage**
  Denna datalagringstjänst kan tillämpas i de fallen där lågkostnad, hög hållbarhet och hög tillgänglighet är viktiga aspekter. Exempel på fall där Nearline Storage kan användas är arkivering av data, online säkerhetskopiering och disaster recovery [24].

2.5.2 Amazon Simple Storage Service (S3)

Amazon tillhandahåller datalagringstjänster till utvecklare och IT organisationer som har hög säkerhet, hög hållbarhet och hög skalbarhet. Med hjälp av ett webbserviceinterface som Amazon erbjuder, blir det enklare att lagra samt hämta data i vilken tidpunkt som helst. Tjänsterna kategoriseras under namnet Amazon S3 och erbjuder kostnadseffektiv lagring för olika typer av fall, som t.ex. säkerhetskopiering, arkivering, disaster recovery, innehållsdistribution. Beroende på vad för typ av datalagringssystem som önskas, så erbjuder Amazon tre datalagringsslösningar, vilka är S3 Standard Storage – General Purpose, S3 Standard Storage – Infrequent Access och Glacier Storage [25].

- **Amazon S3 Standard Storage – General Purpose**
  Denna datalagringstjänst är avsedd för frekvent åtkomst till data och kan tillämpas i de fallen där hög hållbarhet, hög datagenomströmning, hög tillgänglighet samt hög prestanda är ett krav. Exempel på fall där Standard Storage – General Purpose kan användas är t.ex. molnapplikationer, dynamiska webbsidor, innehållsdistribution samt mobila- och spelapplikationer [26].

- **Amazon S3 Standard Storage – Infrequent Access**
  Denna datalagringstjänst är avsedd för åtkomst till data som inte sker ofta och kan tillämpas i de fallen där hög hållbarhet, hög datagenomströmning samt hög prestanda är ett krav. Exempel på fall där Standard Storage – Infrequent Access kan användas är t.ex. långvarig lagring, säkerhetskopiering samt disaster recovery [26].

- **Amazon Glacier Storage**
  Denna datalagringstjänst är avsedd för åtkomst till data som sker mycket sällan och kan tillämpas i de fallen där hög säkerhet, hög hållbarhet samt lågkostnad är ett krav. Ett typiskt fall där Glacier Storage kan användas, är arkivering av obegränsad mängd av data som sällan används [26].

2.5.3 Microsoft Azure Storage

Microsoft tillhandahåller datalagringstjänster som har hög flexibilitet och hyperskala, vilket är väsentligt vid lagring och hämtning av stora mängder av data. Tjänsterna som Microsoft erbjuder, består av olika typer av lagring. Tjänsterna som Microsoft erbjuder, är anpassade till kunder beroende på vilken typ av data som lagras samt till vilket syfte. Detta innebär att kunden har möjlighet att välja graden av pålitligheten för data som ska lagras. Microsoft erbjuder datalagringslösningar i form av Blob Storage [27].

Blob Storage

Denna datalagringsform är användbar vid lagring av dokument och mediafiler, som är exempel på ostrukturerad data. Med hjälp av Blob Storage, så kan kunderna lagra data både kostnads- och skalbarhetseffektivt.

Övriga funktioner som erbjuds inom Blob Storage är att användare kan hantera åtkomst till det lagrade datan samt ge full kontroll över en lagringsstruktur (lämpade för systemadministratörer) [28]. Microsoft erbjuder Blob Storage genom fyra datalagringslösningar, vilka är Locally redundant-storage (LRS), Zone-redundant Storage (ZRS), Geo-redundant Storage (GRS) och Read-access geo-redundant Storage (RA-GRS) [29].

- **Locally redundant-storage (LRS)**
  Denna datalagringstjänst är avsedd för användaren som vill ha ekonomisk lokal lagring, datastyrningskrav samt skapa flera synkrona kopior av data inom ett och samma datacenter [30].

- **Zone-redundant Storage (ZRS)**
  Denna datalagringstjänst är avsedd för användaren som vill ha ekonomisk och blockbaserad lagring samt skapa tre kopior av data på flera datacenter i en eller flera regioner [30].

- **Geo-redundant Storage (GRS)**
  Denna datalagringstjänst är avsedd för användaren som vill ha skydd mot en katastrof i ett datacenter samt skapa flera synkrona kopior till andra datacenter som ligger flera hundratals kilometer bort. Datalagringstjänsten erbjuder samma funktioner som LRS [30].

- **Read-access geo-redundant Storage (RA-GRS)**
  Denna datalagringstjänst är avsedd för användaren som vill ha läsåtkomst av data i ett sekundärt datacenter vid driftstopp genom att erbjuda hög tillgänglighet och tålighet. Tjänsten erbjuder användaren att skapa sex kopior av data och har samma funktioner som GRS [30].

2.6 Användningsfallen för datalagringssystem

Följande avsnitt ger en allmän och en teknisk beskrivning av valda användningsfall, där motivering av valda användningsfall framgår i avsnitt 1.2.

2.6.1 Cold Storage

Cold Storage är ett användningsfall för lagring av data som används sällan. Låg kostnad, hög lagringskapacitet och hög skalbarhet är väsentliga aspekter vid implementation av detta användningsfall. Datahämtning och responstid kan vara betydligt långsammare vid användning av Cold Storage jämfört med andra användningsfall där hög prestanda har högst prioritet. Typiska exempel där Cold Storage har tillämpning är dataarkivering av multimedia där enorma mängder av data i form av videoklipp och bilder laddas upp på olika sociala medier, vetenskapliga undersökningar, där mätdata analyseras och behöver arkiveras för eventuella framtida analyser eller arkivering av hälsojournaler [31, 32]. Bland de kommersiella lösningarna som finns på marknaden just nu är *Amazon S3 (Glacier)* och *Google Cloud Storage (Nearline)*, medan när det gäller distribuerade filsystem som bygger på öppen källkod är de mest intressanta lösningarna tillhandahållna av *Ceph*, *SwiftStack* och *GlusterFS*.

2.6.1.1 Dataarkivering

Dataarkivering är en process där data som inte används ofta, flyttas över till en separat lagringsenhet för långtidsbevaring. Arkiverad data består av information som lagrats länge, har fortfarande betydelse för en tjänsteanvändare, då informationen behöver användas som en referens framöver. Dataarkiveringar är indexerade och användaren ges en möjlighet att söka igenom arkiveringar efter data, så dessa kan enkelt lokaliseras och hämtas [33, 34].

Den stora fördelen med att ha en separat lagringssystem som är gjord för arkivering av data är att det minskar på kostnaderna, då kraven på sådana lagringssystem bygger oftast på lågprestanda och hög mängd av lagringsvolym (hög skalbarhet).

2.6.1.2 Disaster Recovery

Disaster recovery är strategier inom säkerhetskopiering och återställning, som går ut på att lagra och bevara kopior av data i ett lagringssystem. Målet med att ha en disaster recovery är att ett företag ska ha en möjlighet att återhämta data samt kunna verkställa en failover[3] på ett lagringssystem vid tillfällen då det kan ske ett fel i lagringssystemet med hjälp av dataskydd. Exempelvis på fel kan vara hårdvarufel, systemkrasch eller virusattack [35, 36].

[3] **Failover** innebär att ett system går i ett standby-läge när det uppstår ett fel i det primära systemet eller att det primära systemet blir stillastående [37, 38].

2.6.1.2.1 Dataskydd i Ceph

Vid skydd av data använder sig Ceph av replikering. När data ska lagras i klustret, så placeras motsvarande objekt till denna data i olika PG (Placement Group) och skrivs sedan i OSDs (se bilaga 4). PG utgör ett virtuellt kluster av sammankopplade OSDs [39]. Val av PGs och OSDs bestäms av CRUSH algoritmen vid tillgängligt diskutrymme. Inom Cephsystemet finns det tre olika strategier på replikering, vilka är följande [40]:

- **Primary-copy**
  Vid primary-copy replikering, vidarebefordrar den första OSD i PGn till andra OSDs om att en skrivning ska göras. När en bekräftelse har skickats, så kan skrivningarna av data ske i OSDn och därmed är läsning av denna data tillåtet.

- **Chain**
  Vid chain replikering, skrivs data sekventiellt över alla OSDs och läsning av denna data är tillåtet så fort en replikering har gjorts på den sista OSD.

- **Splay**
  Vid splay replikering, skrivs hälften av alla repliker av data sekventiellt och sedan parallellt. Läsning av data är sedan tillåten.

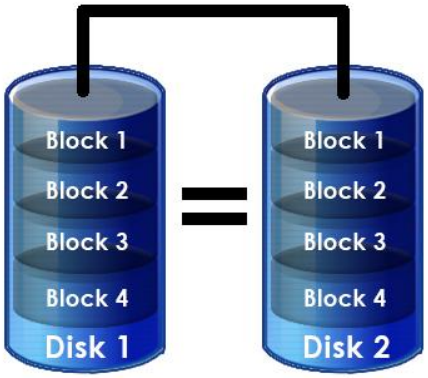
2.6.1.2.2 Dataskydd i SwiftStack

Vid skydd av data använder sig SwiftStack av replikering. När objekt som placeras ut i klustret ska lokaliseras, använder sig SwiftStack av en datastruktur så kallad Ring. Ring håller tre olika referenser för att lokalisera data, vilka är *accounts*, *containers* och *objects*. Vid en I/O-förfrågan, kollar Ring först vilket account efterförfrågad data hör till. Account har i sin tur referens till olika containers, som i sin tur utgör logiska minnesplatser. Vidare håller containers referens till det verkliga minnesutrymmet (på lagringsmedia), där data finns lagrad i form av objekt. Detta är det slutgiltiga steget för att lokalisera data i SwiftStack. Denna datalokaliseringsmetod är väsentligt i SwiftStack för att kontrollera om data behöver replikeras till en disk. Algoritmen som används för att replikera data i ett SwiftStack kluster kallas *unique-as-possible*. Denna algoritm har en egenskap som kollar minst använda lagringsutrymme i hela klustret. Om klustret sträcker sig ut genom geografiskt avlägsna noder, så kollar algoritmen först vilken av de noderna som är minst belastade och replikerar data där. Denna algoritm säkerställer att datareplikerna befinner sig så långt som möjligt från varandra. Detta minskar risken ifall naturkatastrofer skulle inträffa [41].

2.6.1.2.3 Dataskydd i GlusterFS

Vid skydd av data använder sig GlusterFS av en RAID-liknande metod (RAID 1). Detta innebär att flera kopior av en hel lagringsenhet görs till andra lagringsenheter inom samma volym genom att använda sig av synkrona skrivningar. Synkrona skrivningar betyder att en volym består av flera delmängder, där det finns en inledande lagringsenhet och dess speglar [40].

I detta fall, måste antalet av lagringsenheter vara en multipel av det önskade antalet repliker (se figur 8). Ett exempel på detta är att det existerar fyra lagringsenheter och replikering önskas att vara två. Den första lagringsenheten kommer att replikeras till den andra lagringsenheten och skapa en delmängd. På samma sätt, så kommer den tredje lagringsenheten att replikeras till den fjärde lagringsenheten och skapa ytterligare en delmängd [40].



Figur 8: Figuren visar en illustration på hur RAID 1 fungerar, där den första lagringsenheten (block 1) i disk 1 replikeras till den andra lagringsenheten (block 1) i disk 2 och på så viss skapar en spegelbild av sig.

2.6.1.3 Commodity Hardware

I ett IT-sammanhang innebär commodity hardware en hårdvaruenhet eller en komponent av en hårdvaruenhet som är relativt billig, allmänt tillgänglig och mer eller mindre utbytbar med en annan hårdvara av samma typ. Att den är utbytbar i denna mening innebär att den är brett kompatibel och att den har förmåga att fungera som plug and play med en annan hårdvara. Dessa är de viktigaste egenskaperna som en commodity hardware kännetecknas av [42, 43].

2.6.2 High Performance Storage

High Performance Storage är ett användningsfall som är bäst lämpad för lagringssystem som kräver låg latens vid dataåtkomst eller lagringssystem vars dataåtkomst är frekvent (“hot” objects). I detta användningsfall betraktas prestanda som högsta prioritet därmed kostnad per GB/månad blir högre i jämförelse med *Cold Storage*. Typiska exempel på *High Performance Storage* är mobila- och onlinespelapplikationer, diverse videoströmningstjänster som YouTube eller musikströmningstjänster som Spotify, dynamiska hemsidor. Ett sådant system bör karakteriseras av skalbarhet på prestandanivå. Detta innebär att om antal användare som tjänsteleverantören betjänar skulle öka, skulle inte systemprestanda påverkas. Nämligen prestandan bör kunna bibehållas samtidigt som belastningsintensiteten på systemet växer [44].

Bland de kommersiella lösningarna som finns på marknaden just nu är *Amazon S3 (Standard)* och Google *Cloud Storage (Standard)*, medan när det gäller distribuerade filsystem som bygger på öppen källkod är de mest intressanta lösningarna tillhandahållna av *Ceph* och *GlusterFS*.

2.6.3 Virtual Machine Storage

Virtual Machine Storage är ett användningsfall som möjliggör att multipla virtuella maskiner körs på en och samma hårdvara. Lagringsystem kan virtualiseras på olika sätt så att prestandaisolering, skalbarhet och effektivt lagringsutnyttjande uppnås [45, 46].

Viktiga egenskaper inom Virtual Machine Storage är snapshotting, thin provisioning, multitenancy. En aspekt som är väsenligt i Virtual Machine Storage är prestandaisolering (performance isolation) vilket innebär att användare får jämna resurser, exempelvis IOPS [47]. Även dataisolering (multitenancy) är en viktig aspekt, vilket innebär att användare inte ska få åtkomst till varandras data [47]. Bland de kommersiella lösningarna som finns på marknaden just nu är *VMware*, *EMC²* och *IBM (Spectrum Storage)* medan när det gäller distribuerade filsystem som bygger öppen källkod är de mest intressanta lösningarna tillhandahållna av *Ceph*.

2.6.3.1 Snapshotting

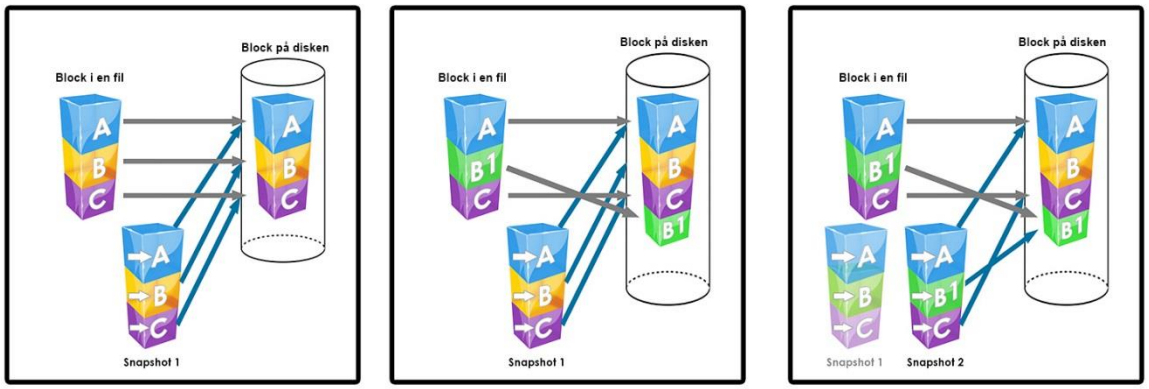
Snapshotting eller ögonblicksbild, beskrivs som en alltmer nödvändig faktor inom datalagring. Datalagringsleverantörer betraktar snapshot som ett medel att skydda data på ett effektivt sätt i sina virtuella miljöer. Vidare definieras snapshot som en ögonblickskopia av filsystemet, som en ”frusen”, sekundär instans av data som kan användas för säkerhetskopiering, datareplikering och även som en grund för att sätta upp en ny virtuell miljö [48, 49].

De flesta fil- och lagringssystem har en dubbelskiktad ordning för att hantera lagrad data. Första skiktet består av metadata, som i sin tur är en katalog av pekare som pekar på det andra skiktet, som i sin tur är den faktiska datapositionen på hårddisken. Istället för att kopiera hela skikt 2, med hjälp av ett snapshot tas det en kopia på skikt 1, alltså endast metadata kopieras. Denna kopia tas nästan ögonblickligen och den upptar betydligt mindre lagringsutrymme jämfört med om kopia på hela filsystemet skulle tagits. De blocken på hårddisken där snapshotkopian finns sätts då till *read-only*, vilket innebär att ingen skrivning till de blocken är möjlig utan bara läsning från dessa. Detta i sin tur säkerställer att data inte manipuleras och därmed förstörs. Efter att en snapshotkopia har gjorts, upprätthålls två exakta kopior av metadata, en aktiv och en passiv eller statisk kopia.

Systemet fortsätter att uppdatera den aktiva kopian, medan den statiska kan användas för säkerhetskopiering eller datareplikering. Vidare beskrivs det att två metoder av snapshotting finns, vilka är copy-on-write och clone or split-mirror [48, 49].

2.6.3.1.1 Copy-on-write

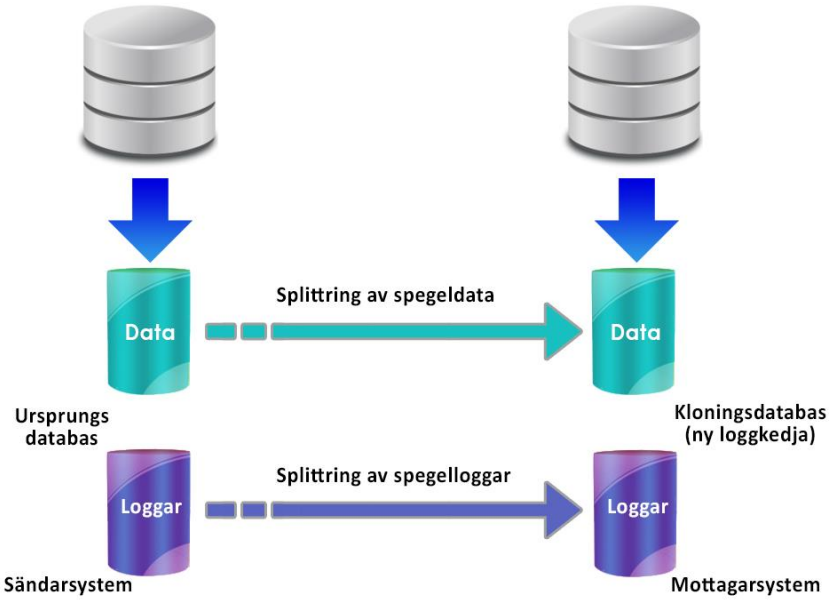
Copy-on-write metoden går ut på att det reserverade diskutrymmet inte behöver vara så stort. Nämligen snapshot-kopior sparar endast metadata och inte det faktiska data som metadata ”pekar” på. De ändringar som görs efter att ett snapshot har skapats skrivs till ett nytt block och pekaren riktas då mot det nya blocket (se figur 9). Alltså, snapshot-kopior ”följer” endast de uppdateringar som gjorts sedan den föregående snapshot-kopian skapats. Tack vare denna egenskap skapas snapshot-kopior nästan ögonblickligen, medan systemet är igång. Copy-on-write metoden anses vara väldigt utrymmeseffektiv, vilket är en stor fördel för denna metod. Denna egenskap är viktig eftersom enbart förändringarna som läses av och skrivs till ett nytt block vid snapshotting, är oftast minimala och kräver därmed inte mycket lagringsutrymme. Nackdelen med denna metod är att prestandan av den ursprungliga volymen minskas. Detta för att skrivförfrågningar till den ursprungliga volymen måste vänta med att köras färdigt tills det ursprungliga data har kopierats till snapshotten [49].



Figur 9: Figuren visar en illustration av hur en snapshotkopia skapas med hjälp av copy-on-write. I vänstra delen av figuren visas då ett snapshot tas. I mittersta delen av figuren skrivs de ändringar till ett nytt block och pekaren uppdateras, men snapshot-pekaren pekar fortfarande till det gamla blocket och därmed en levande kopia över data bevaras. I den högra delen av figuren tas en annan snapshot-kopia. Nu finns det tre kopior av data utan att det tas upp extra diskutrymme som tre separata kopior skulle göra.

2.6.3.1.2 Clone or split-mirror

Clone or split-mirror metoden går ut på att en fysisk klon skapas av en datamängd, ett filsystem eller ett LUN. På så sätt skapas en snapshot-kopia av det faktiska data av både samma storlek och samma typ och placeras på en annan fysisk enhet som en säkerhetskopia (se figur 10). Kloner utgör då exakta kopior av den ursprungliga datavolymen. Dock är skapelse av sådana snapshot-kopior inte ögonblicklig då det är hela fysiska volymen som kopieras, samt att kopieringstiden starkt beror av volymstorleken som kopieras [27]. Clone or split-mirror metoden kännetecknas av hög tillgänglighet, vilket är en stor fördel för denna metod. Denna egenskap är uppfylld eftersom kloner som skapas från den ursprungliga datavolymen, är exakta kopior. Nackdelen med denna metod är att den kräver mycket lagringsutrymme. Detta för att varje snapshot-kopia som skapas, tar lika mycket utrymme som själva datamängden som görs en snapshot-kopia. En annan nackdel med denna metod är att prestandan påverkas kraftigt beroende på datamängd som ska kopieras samtidigt som systemet är igång [49].



Figur 10: Figuren visar illustration av hur en snapshotkopia skapas med hjälp av clone or split-mirror, där hela datablocket (se turkosa cylindrar) kopieras till ett nytt block. Vid nya ändringar, kopieras hela datablocket igen (se lila cylindrar) ytterligare till ett nytt block. Denna metod är mest lämplig för säkerhetskopiering.

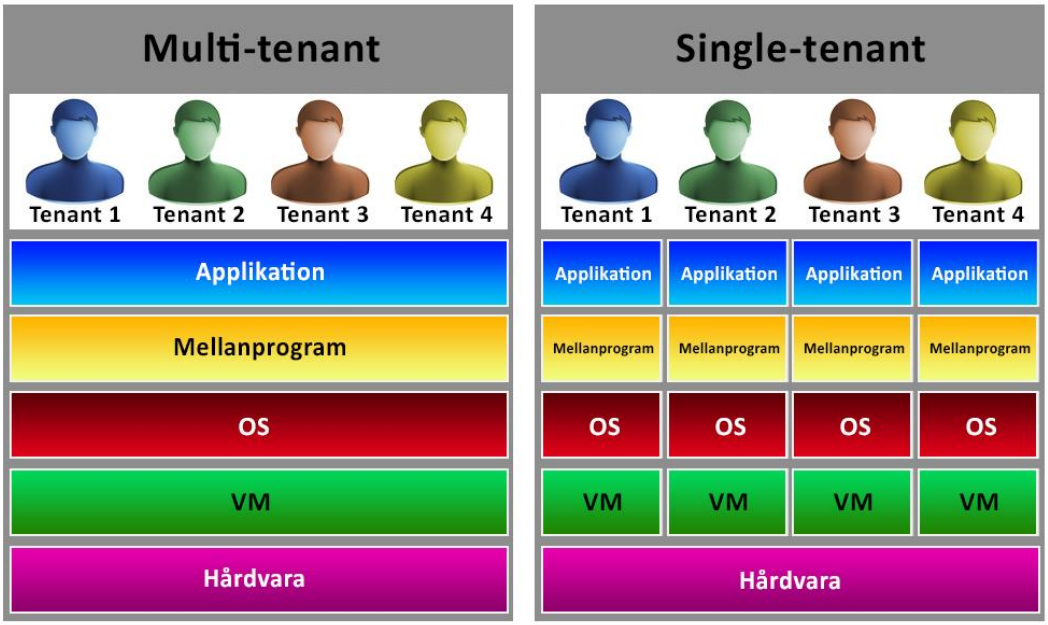
2.6.3.2 Isolering i datalagringssystem

Isolering i ett datalagringssystem kan vara i form av dataisolering (skydd av data) eller prestandaisolering (där prestanda inte påverkas). Följande avsnitt ger en beskrivning av dessa typer av isolering.

2.6.3.2.1 Multi-tenancy (dataisolering)

I arbetet ”Architectural Concerns in Multi-Tenant SaaS Applications” beskrivs multi-tenancy ur tjänsteleverantörernas perspektiv. Multi-tenancy är en mjukvaruarkitektur som bygger på att resurs (applikationsinstans) delas mellan olika användare (tenants[4]), där alla användare får varsin del av instansen (se figur 11). Detta innebär att användare är isolerad från andra delar med hänsyn till prestanda och enskildhet [47].

En analogi med multi-tenancy kan vara en grupp bestående av några personer som bor i ett och samma hus och delar kostnader för värmen men som å andra sidan vill ha sin enskildhet, och därför har de varsitt rum (isolering) för att exempelvis undvika buller [47].



Figur 11: Figuren visar en illustration på hur multi-tenant och single-tenant arkitektur fungerar. Vid en multi-tenant arkitektur, delar alla tenants samma applikation, mellanprogram, OS, VM och hårdvara. Vid en single-tenant arkitektur, har varje tenant sitt egen applikation, mellanprogram, OS och VM men delar samma hårdvara.

[4] **Tenant** är en grupp av olika användare som delar samma applikationsinfrastruktur. Detta innebär tillgång, konfigureringar, user management och viss applikationsfunktionalitet med hänsyn till dataskydd och enskildhet [47].

2.6.3.2.2 Performance Isolation (prestandaisolering)

Vidare i arbetet ”Architectural Concerns in Multi-Tenant SaaS Applications”, beskrivs även performance isolation med hänsyn till mätt prestanda och den kvoten som varje användare har till sitt förfogande. I detta fall så innebär kvoten ett mått på hur mycket en användare får förbruka en delad resurs. Försämring av prestanda är ett av de största hinder för eventuella användare av molntjänster. De prestandarelaterade problemen orsakas oftast genom att en andel användare överstiger sin kvot [47]. Det beskrivs även olika nivåer av prestandaisolering, vilka är följande:

- **Prestandaisolerat system**
  I detta fall tillåts olika användare att överstiga sin kvot utan att andra användares prestanda påverkas, det vill säga den förbestämda kvoten tillåtet av SLA kan fortfarande utnyttjas maximalt oavsett de andra användarnas utnyttjandegrad. Ett exempel på ett prestanda isolerat system är hur CPU utnyttjas vid processhantering i ett datorsystem [47].

- **Svagt prestandaisolerat system**
  I detta fall får vissa användare överstiga sin kvot så länge de inte stör andras prestanda. Om vissa användare skulle överstiga sin kvot på andra användares bekostnad, skulle systemet ha möjlighet att begränsa kvoten till den förbestämda nivån tillåtet av SLA för de användare som överutnyttjar systemet [47].

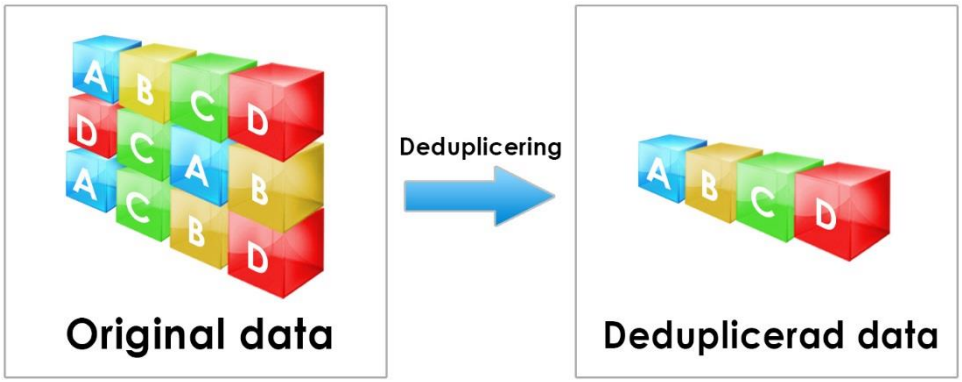
- **Oisolerat system**
  I detta fall garanteras inte någon av ovannämnda egenskaperna. Detta innebär att användare som arbetar inom sin kvot kommer direkt att påverkas av en användare som överstiger sin kvot [47].

2.6.3.3 Deduplicering av data

Deduplicering av data är en metod för att minska lagringsutnyttjande genom att minska redundant data. Ett exempel kan tas för ett typiskt emailsystem där systemet består av hundra instanser av ett och samma bifogade fil av storleksordning 1 MB. Ifall systemet skulle behövas säkerhetskopieras så skulle de hundra instanserna kräva 100 MB av diskutrymme. Med hjälp av deduplicering, kopieras endast en bifogad fil och alla andra underinstanser refererar till den kopierade filen (se figur 12). Därmed reduceras lagringsutnyttjande från 100 MB till 1 MB genom att endast en kopia av det redundanta data lagras [50, 51].

Det finns fördelar som gynnar de tjänsteleverantörer som implementerar deduplicering. Dessa är följande [52, 53]:

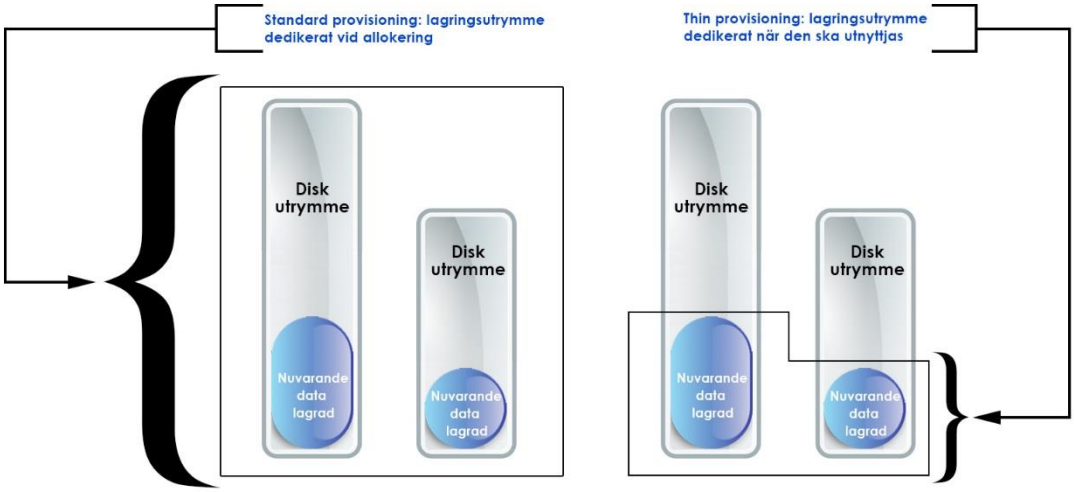
- ✓ Minskade hårdvarukostnader, i och med att mindre datamängder som behöver lagras upptar mindre lagringsutrymme och därmed sparas lagringsutrymme.
- ✓ Minskade säkerhetskopieringskostnader, om mindre datamängder behöver sparas innebär automatiskt att mindre datamängder behöver säkerhetskopieras i fall det är nödvändigt.
- ✓ Minskade kostnader för katastrofåterställning.
- ✓ Ökad systemprestanda och nätverksprestanda, mindre lagrade datamängder innebär mindre data som ska säkerhetskopieras vilket i sin tur innebär mindre data som ska skickas över via nätet och därmed blir mindre belastning på nätverket.



Figur 12: Figuren visar en illustration på hur deduplicering av data fungerar med hjälp av block. Exempelvis skulle block A vara 1 MB och skulle ta 3 MB av lagringsutrymmet. Vid deduplicering, minskas storleken på data till 1 MB.

2.6.3.4 Thin provisioning

Thin provisioning är en metod för att öka effektivitet av användning på tillgängligt diskutrymme i SAN. Thin provisioning ser till att det dedikeras en mängd av diskutrymme för en del användare, baserade på det diskutrymmet som krävs för dessa användare vid olika tillfällen (se figur 13). Exempelvis på tillfällen kan vara mängden av data som växer, vilket leder till att mer utrymme krävs. Då kan det dedikerade diskutrymmet för användarna ökas och detta eliminerar problemet med att betala för en stor mängd av diskutrymme som inte kommer att användas fullkomligt vid senare tillfälle [54, 55].



Figur 13: Figuren visar en illustration på hur ett standard provisioning och thin provisioning ser ut i ett lagringssystem. Vid standard provisioning, dedikeras en mängd av lagringsutrymmet i förväg. Detta kan observeras med hjälp av det fyrkantiga mönstret (se stora klammern till vänster) som täcker hela lagringsutrymmet. Vid thin provisioning, dedikeras en mängd av lagringsutrymmet när den ska utnyttjas. Detta kan observeras med hjälp av det rektangulära mönstret (se lilla klammern till höger) som täcker enbart det lagringsutrymmet som ska utnyttjas.

3 Metod och resultat

Detta kapitel handlar om analys och jämförelser som utfördes i form av undersökning av mätningar som gjorts i tidigare arbeten samt lyfter upp de viktigaste egenskaperna hos de omnämnda öppna distribuerade filsystem har (se avsnitt 2.4). Metodiken som användes för att utföra undersökningen, bygger på utvärderingar baserade på analys och jämförelser av tester från tidigare arbeten. Denna metodik kan dra med sig problem som kan vara avgörande då det slutgiltiga resultatet ska analyseras och sammanfattas och därmed försvåra fastställande av en slutgiltig lösning. Ett tydligt exempel kan vara jämförelse av två prestanda tester som har utförts i två olika arbeten. Denna jämförelse kan betraktas opassande då dessa tester utfördes under olika omständigheter bland annat olika hårdvaruinfrastrukturer, olika mättider eller olika versioner av samma programvara.

Då det redan har nämnts i avgränsningarna att kommersiella lösningar enbart ska undersökas för att ge en överblick över distribuerade filsystem, har störst fokus lagts endast på de öppna distribuerade filsystem som nämns i avsnitt 1.3. Detta för att kunna ta bestämma vilket öppet distribuerat filsystem som är mest lämpligt för de olika användningsfallen gällande kostnad, prestanda samt dataskydd. I denna undersökning utfördes studier av de kommersiella- och öppna DFS (som nämns i avsnitt 2.4 och 2.5) utifrån litteratur, tidigare arbeten inom datalagring samt tillgänglig fakta kring koncept som är relevanta till datalagring.

Kapitlet är indelat på följande sätt:

Första delen av kapitlet ger en sammanställning av mätningar, jämförelser och egenskaper som har gjorts från tidigare arbeten för kommersiella DFS (avsnitt 3.1 och 3.2). Detta för att ge en översikt av hur de etablerade företagens datalagringslösningar fungerar när det gäller prestanda, kostnader och dataskydd.

Den större delen av kapitlet utgörs av en sammanställning av mätningar och egenskaper som gjorts från tidigare arbeten för öppna DFS (avsnitt 3.3 och 3.4). Denna del ger en mer detaljerad analys och jämförelse av de mätningar och egenskaper som tagits fram för att kunna dra en slutsats om vilket öppen distribuerat filsystem som företaget AxByte kan implementera.

En kostnadsuppskattning för ett kommersiellt- och öppet distribuerat filsystem har även gjorts (avsnitt 3.5). Detta för att kunna få en översikt över hur den totala kostnaden blir ifall ett eget distribuerat filsystem (öppet) ska användas eller ifall en datalagringslösning ska köpas av ett större företag (kommersiellt).

Kapitlet avslutas med en sammanfattning och summering av resultaten som åstadkommits vid undersökning av kommersiella- och öppna DFS (avsnitt 3.6).

3.1 Prestandamätningar och kostnadsjämförelser utförda på kommersiella distribuerade filsystem

I avsnittet presenteras först prestandamätningar, följt av kostnadsjämförelser utifrån tidigare arbeten. I slutet av avsnittet presenteras ytterligare egenskaper hos samtliga kommersiella distribuerade filsystem utifrån faktainsamling. Dessa kommersiella lösningar som användes i nedanstående arbeten gäller endast för objektbaserad lagring. Därmed nya lösningar med andra prisplaner kommer att introduceras längre fram, då den slutgiltiga kostnadsuppskattningen för kommersiella- och öppna DFS görs.

3.1.1 Läsning och skrivning

Följande avsnitt beskriver de prestandamätningar som gjordes för kommersiella distribuerade filsystem gällande läsning och skrivning.

3.1.1.1 Mätning av läs- och skrivtester mellan Google, Microsoft och Amazons datalagringslösningar

I arbetet ”Vklass datalagring” utfördes läs- och skrivtester mellan Googles, Microsofts och Amazons datalagringslösningar [6]. Testerna mätte prestanda i form av datahastighet (MB/s).

Ett uppladdning- och nedladdningstest utfördes med 10 filer, där varje fil var 1 MB. Detta innebar en total storlek på 10 MB. Testerna mätte genomsnittlighastighet utifrån åtta mätningar under ett dygn och gav följande resultat (se tabell 1, 2, 3, 4, 5, 6 och 7).

Tabell 1: Tabellen visar den genomsnittliga uppladdning- och nedladdningshastighet för Google Standard Storage (se sidan 53) [6].

Skriv	Läs
5,73 MB/s	6,08 MB/s

Tabell 2: Tabellen visar den genomsnittliga uppladdning- och nedladdningshastighet för Google DRA Storage (se sidan 53) [6].

Skriv	Läs
3,01 MB/s	6,14 MB/s

Tabell 3: Tabellen visar den genomsnittliga uppladdning- och nedladdningshastighet för Google Nearline Storage (se sidan 53) [6].

Skriv	Läs
4,31 MB/s	2,99 MB/s

Tabell 4: Tabellen visar den genomsnittliga uppladdning- och nedladdningshastighet för Amazon S3 Standard Storage – General Purpose (se sidan 54) [6].

Skriv	Läs
5,79 MB/s	7,12 MB/s

Tabell 5: Tabellen visar den genomsnittliga uppladdning- och nedladdningshastighet för Amazon S3 Standard Storage – Infrequent Access (se sidan 55) [6].

Skriv	Läs
4,31 MB/s	5,77 MB/s

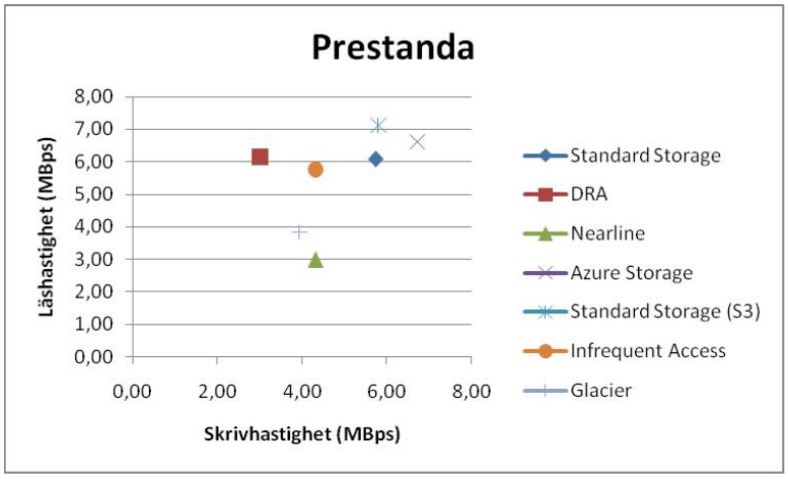
Tabell 6: Tabellen visar den genomsnittliga uppladdning- och nedladdningshastighet för Amazon Glacier Storage (se sidan 55) [6].

Skriv	Läs
3,93 MB/s	3,84 MB/s

Tabell 7: Tabellen visar den genomsnittliga uppladdning- och nedladdningshastighet för Microsoft Azure Storage (se sidan 54) [6].

Skriv	Läs
6,71 MB/s	6,61 MB/s

Figur 14 sammanställer resultat som ovanstående tabeller visar i en graf.



Figur 14: Figuren visar en sammanställning av samtliga läs- och skrivtester för Googles, Microsofts och Amazons datalagringslösningar. På x-axeln representeras skrivhastigheten och på y-axeln representeras läshastigheten. *Bildkälla:* [6], se sidan 59.

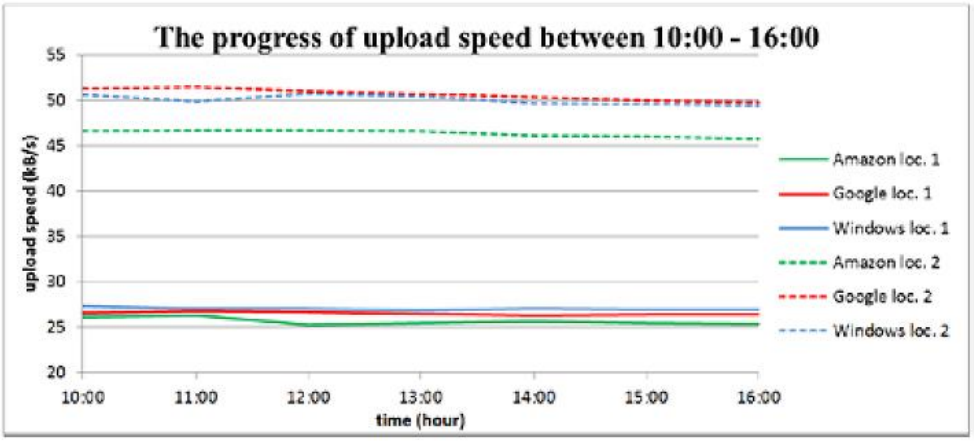
Kommentar

Ur figur 14 kan det observeras att olika datalagringslösningar varierade i läs- och skrivprestanda. Amazon Standard visade bäst läshastighet och Microsoft Azure Storage visade bäst skrivhastighet.

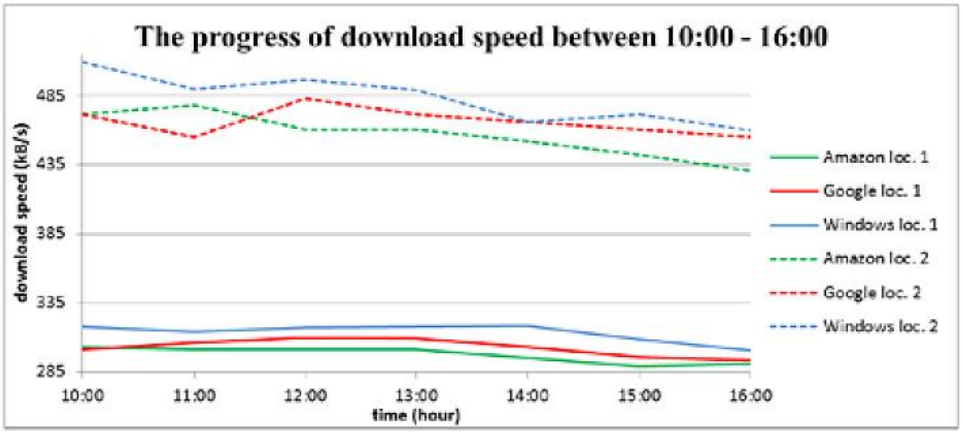
3.1.1.2 Mätning av läs- och skrivtester mellan Google, Microsoft och Amazon

I arbetet ”Performance Testing of Cloud Storage while Using Spatial Data” utfördes läs- och skrivtester mellan Google, Microsoft och Amazon [56]. Testerna mätte prestanda i form av datahastighet (kB/s).

Testerna utfördes på två olika platser då syftet var att ta reda på hur nätverkshastigheten på geografiskt olika platser kan påverka prestanda. Ett uppladdning- och nedladdningstest utfördes med en fil av storleksordningen 3,57 MB. Testerna mätte genomsnittlighastighet varje timme mellan 10:00 och 16:00. Följande figurer (se figur 15 och 16) visar erhållna resultat för läs- och skrivtesterna. Streckade kurvor föreställer mätningar från ena mätplatsen (plats A) och heldragna kurvor föreställer mätningar från den andra mätplatsen (plats B). För mer detaljerat resultat av datahastigheterna, se bilaga 5.



Figur 15: Figuren visar den genomsnittliga uppladdningshastigheten mellan 10:00 och 16:00 för Google, Microsoft och Amazon på två olika platser. *Bildkälla:* [56], se sidan 4.



Figur 16: Figuren visar den genomsnittliga nedladdningshastigheten mellan 10:00 och 16:00 för Google, Microsoft och Amazon på två olika platser. *Bildkälla:* [56], se sidan 4.

Kommentar

Ur ovanstående figurer, kan observeras att prestanda mellan de olika datalagringslösningarna inte varierade så mycket. Dock de nedladdningstesterna som utfördes på mätplatsen A visade avsevärda prestandaavvikelser, vilket kunde bero på nätverkshastigheten.

3.1.2 Kostnadsjämförelser

Följande avsnitt beskriver de kostnadsjämförelser som gjordes för kommersiella distribuerade filsystem.

3.1.2.1 Jämförelse av kostnader mellan Googles, Microsofts och Amazons datalagringslösningar

I arbetet ”Vklass datalagring” jämfördes kostnader mellan Googles, Microsofts och Amazons datalagringslösningar [6]. Kostnaderna jämfördes i form av SEK per GB/månad.

Kostnaderna gällde per GB och månad och presenteras i följande tabeller (se tabell 8, 9 och 10). Valutakursen som användes för omvandling av priserna till svenska kronor, togs den 11 juli 2016.

Tabell 8: Tabellen visar priserna för datalagringslösningarna som Google erbjuder (se sidan 41 och 42) [6].

Datalagringslösning	**Pris per GB/månad (USD)**	**Pris per GB/månad (SEK)**
Google Standard Storage	0,026	≈ 0,22
Google DRA Storage	0,02	≈ 0,17
Google Nearline Storage	0,01	≈ 0,086

Tabell 9: Tabellen visar priserna för datalagringslösningarna som Microsoft erbjuder för Blob storage (se sidan 44) [6].

Datalagringslösning	**Pris per GB/månad (USD)**	**Pris per GB/månad (SEK)**
Locally redundant Storage (LRS)	0,0219	≈ 0,19
Zone-redundant Storage (ZRS)	0,027	≈ 0,23
Geo-redundant Storage (GRS)	0,043	≈ 0,37
Read-access Geo-redundant Storage (RA-GRS)	0,0548	≈ 0,47

Tabell 10: Tabellen visar priserna för datalagringslösningarna som Amazon erbjuder (se sidan 46 och 47) [6].

Datalagringslösning	**Pris per GB/månad (USD)**	**Pris per GB/månad (SEK)**
Amazon S3 Standard Storage – General Purpose	0,0319	≈ 0,28
Amazon S3 Standard Storage – Infrequent Access	0,018	≈ 0,16
Amazon Glacier Storage	0,0120	≈ 0,10

3.1.2.2 Jämförelse av kostnader mellan Google, Microsoft och Amazon

I arbetet ”Performance Testing of Cloud Storage while Using Spatial Data” jämfördes kostnader mellan Google, Microsoft och Amazon [56]. Kostnaderna jämfördes i form av SEK per GB/månad.

Kostnaderna gällde per GB och månad och presenteras i nedanstående tabell (se tabell 11). Valutakursen som användes för omvandling av priserna till svenska kronor, togs den 11 juli 2016.

Tabell 11: Tabellen visar priserna för Amazon, Google och Microsoft (se sidan 2) [56].

Datalagringstjänst	**Pris per GB/månad (USD)**	**Pris per GB/månad (SEK)**
Amazon S3 Storage	0,125	≈ 1,08
Google Cloud Storage	0,12	≈ 1,03
Microsoft Azure Storage	0,125	≈ 1,08

3.2 Övriga egenskaper hos kommersiella distribuerade filsystem

Övriga egenskaper (exempelvis dataskydd) som inte gick att mäta, presenteras i detta avsnitt för kommersiella distribuerade filsystem utifrån faktainsamling. En kortfattad beskrivning av dataskydd från samtliga datalagringstjänster ges nedan.

- **Google Cloud Storage**
  Enligt Google, replikeras data över flera olika geografiskt avlägsna plaster och därmed ökas dataskyddsgraden. Användaren ges möjlighet att öka graden av dataskydd genom att själva lagra kopior av data i flera logiska lagringsutrymmen istället. Därmed minskas sannolikheten att data försvinner då en naturkatastrof inträffar [57].

- **Amazon S3 Storage**
  Amazon skriver på sin hemsida att replikering av data sker över flera lagringsnoder inom Amazons datacenter. Därmed uppnås hög tillgänglighet samt dataskydd. Användaren ges möjlighet att öka dataskyddsgraden genom att välja lagringsalternativet Glacier [58].

- **Microsoft Azure Storage**
  Microsoft erbjuder ett brett sortiment av olika datareplikeringsalternativ där användaren erbjuds möjlighet för upp till tre kopior av data utspridda genom olika avlägsna datacenter. Exempelvis erbjuder LRS tre synkrona kopior av data inom samma datacenter, medan ZRS erbjuder tre synkrona kopior utspridda genom olika datacenter. Extra dataskydd kan uppnås då det finns möjlighet att välja att ZRS även kopierar data genom olika regioner [30].

3.3 Prestanda- och dataskyddsmätningar utförda på öppna distribuerade filsystem

I följande avsnitt presenteras mätningar angående skalbarhet för dataarkivering samt dataskydd. Därefter presenteras mätningar gällande läsning- och skrivning och prestandaisolering utifrån tidigare arbeten. I slutet av avsnittet presenteras ytterligare egenskaper hos samtliga öppna distribuerade filsystem utifrån faktainsamling.

3.3.1 Dataarkivering

Följande avsnitt beskriver de mätningar som utfördes gällande dataarkivering.

3.3.1.1 Mätning av dataarkivering i SwiftStack och Ceph

I arbetet “POSIX and Object Distributed Storage Systems Performance Comparison Studies With Real-Life Scenarios in an Experimental Data Taking Context Leveraging OpenStack Swift & Ceph”, jämfördes arkitektur, prestanda, tillförlitlighet och skalbarhet mellan två objektbaserade lagringssystem, tillhandahållna av SwiftStack och Ceph samt en undersökning av POSIX filsystemet som Ceph erbjuder [59]. I detta examensarbete togs hänsyn endast till mätningar gällande prestanda och skalbarhet. Testerna mätte prestanda i form av datahastighet (MB/s) samt skalbarheten i form av hur snabbt ett distribuerat filsystem kan växa utan att prestanda försämras.

Infrastrukturen där experimentet utfördes på, bestod av flera hundra noder, varav 30 av de implementerade Scientific Linux[5].

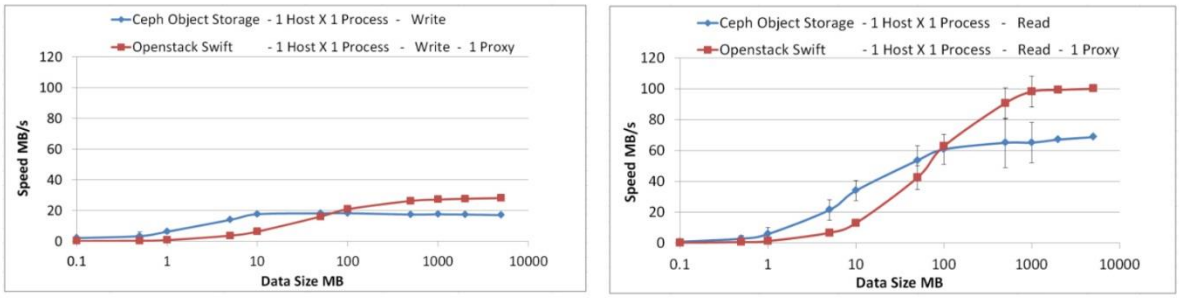
Följande tester gjordes i infrastrukturen:

- **Single host, single stream test (prestandamätning)**
  I detta test utfördes läs- och skrivmätningar för en host (se figur 17).

- **Single host, multiple IO stream test (prestanda- och skalbarhetsmätning)**
  I detta test utfördes läs- och skrivmätningar för en host med 10 parallella skriv- och läsprocesser (se figur 18).

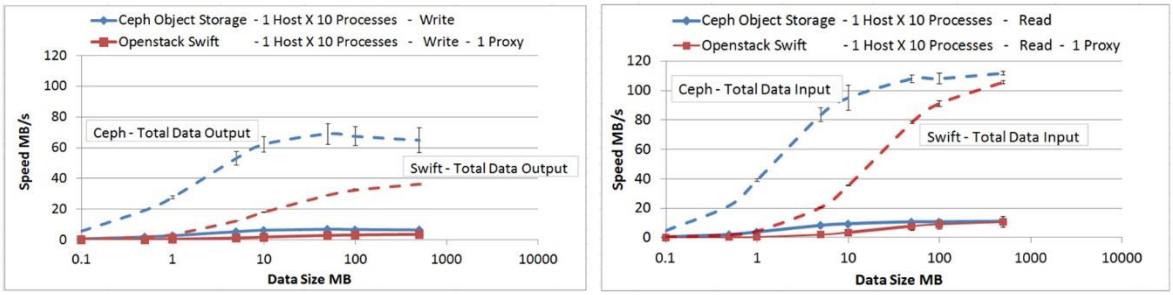
- **Multiple host tests (skalbarhetsmätning)**
  I detta test utfördes skrivmätningar då flera hosts kördes parallellt (se figur 19).

[5] **Scientific Linux** är en Linuxversion skapad av Fermilab och CERN [60].



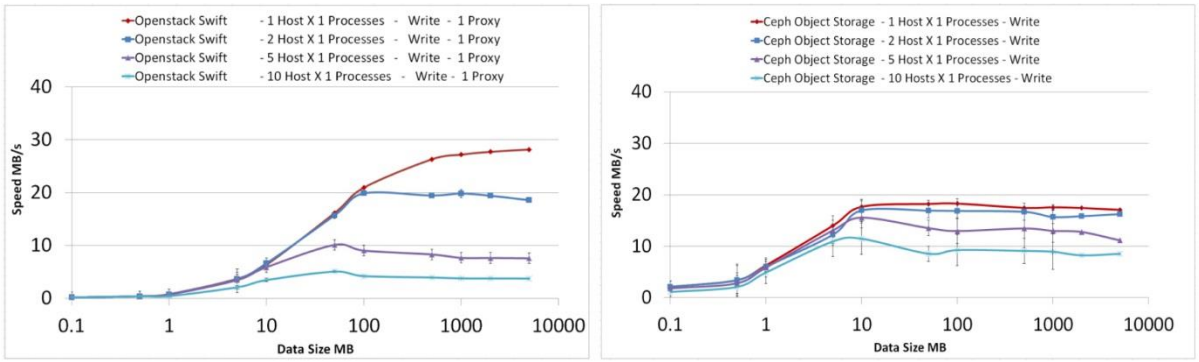
Figur 17: Figuren visar mätningar för läs- och skrivtester för en host som kör en process, där den vänstra delen visar skrivtestet och den högra delen visar lästestet. *Bildkälla:* [59], se sidan 3.

I testet, konfigurerades SwiftStack så att en replik lagrades som en fil enligt standardobjektstorlek 5 GB [61]. Standardobjektstorlek för lagring av filer i Ceph var 4 MB [62]. Dessa fakta var avgörande i mätningen som visades i figur 17. Skrivmätningarna (vänstra delen av figuren) visade att då filstorleken som lagrades översteg 4 MB, började Cephs datahastighet avta vid 10 MB. Detta berodde på att filerna som lagrades behövde delas upp i 4 MB segment, vilket bromsade upp systemet. Dock visade mätningarna att vid läsning av filer presterade SwiftStack avsevärt bättre vid läsning av filer som översteg 100 MB.



Figur 18: Figuren i vänstra delen visar mätning av 10 parallella skrivprocesser medan den högra delen visar mätning av 10 parallella läsprocesser. Heldragna linjerna representerar datahastighet per individuell skriv- och läsprocess medan de streckade linjerna representerar den sammanlagda datahastigheten av alla 10 processerna. *Bildkälla:* [59], se sidan 4.

I detta test (se figur 18) utfördes prestanda- och skalbarhetsmätningar som en funktion av 10 parallella läs- och skrivprocesser. Resultaten som erhölls i denna mätning visade att Ceph uppnådde bättre prestanda då flera läs- och skrivprocesser kördes parallellt, tack vare sin unika arkitektur (se bilaga 1 för arkitekturen på Ceph), vilket möjliggjorde att varje individuell process kunde skapa sin egen kanal till respektive lagringsnod. Detta i sin tur spred ut läs- och skrivförfrågningar genom hela klustret och därmed fler noder. Enligt författarna, ju fler OSD noder som lades till i Ceph klustret desto bättre prestanda borde uppnås. Vidare skrev författarna att Ceph visade bättre prestanda i detta test även på grund av att SwiftStack förde in alla processer genom en proxynod (se bilaga 3 för arkitekturen på SwiftStack), vilket i sin tur skapade flaskhalseffekt då parallella processer kördes.



Figur 19: Figuren visade mätningar av skalbarhet då multipla hosts installerades för skrivtester. Färgerna representerade olika antal hosts, där varje host betjänade en process. *Bildkälla:* [59], se sidan 4.

Det framgick av figur 19 att SwiftStack visade bättre prestanda då en host och en process kördes. Då flera hosts lades till, försämrades SwiftStacks prestanda avsevärt. Enligt mätningarna som observerades i figuren övergick SwiftStacks datahastighet från ca 28 MB/s till ca 4 MB/s då antalet hosts som kördes samtidigt ökade till 10, vilket i sin tur innebar en datahastighetsminskning med ca 87 %. I högra delen av bilden visades Cephs prestanda under samma omständigheter. Av figuren framgick att systemets prestanda minskades med 50 % då antalet hosts ökade till 10 (från ca 18 MB/s till ca 9 MB/s). Enligt författarna ansågs Ceph ha betydligt bättre skalbarhet jämfört med SwiftStack, vilket tydligt visades av figur 19.

Kommentar

Testerna visade att två objektbaserade lagringssystem har liknande prestanda då enstaka filer skrevs. Dock uppnådde Ceph sin maxgräns vid skrivning av stora filer vilket framgick av figur 17. Vidare skrev författarna att vid ökat antal parallella processer visade Ceph bättre prestanda än SwiftStack samt en optimalare datadistribution genom klustret.

3.3.1.2 Mätning av dataarkivering i GlusterFS

I arbetet “Characterizing the GlusterFS Distributed File System for Software Defined Networks Research”, utfördes framförallt tester för skalbarhet men även prestandatester för det objektbaserade lagringssystemet GlusterFS [103]. I detta examensarbete togs hänsyn endast till mätningar gällande skalbarhet och prestanda. Testerna mätte skalbarheten i form av hur snabbt ett distribuerat filsystem kan växa utan att prestanda försämras samt prestanda i form av datahastighet (MB/s).

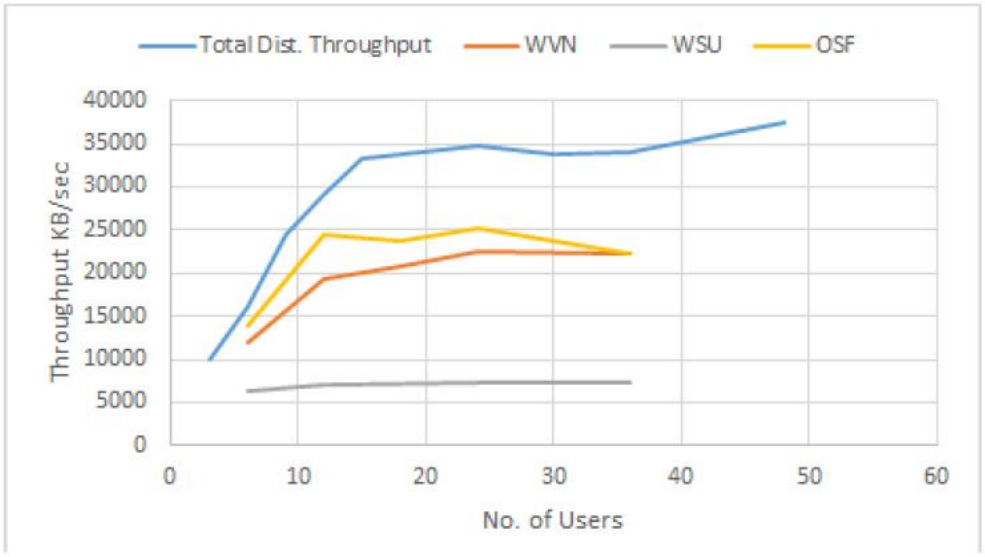
Infrastrukturen där experimentet utfördes på, bestod av sex servrar utspridda genom sex olika städer i USA, varav tre av servrarna användes som lagringsservrar. Resterande servrar användes som GlusterFS klientservrar (OSF, WVN och WSU). För en mer detaljerat översikt av infrastrukturen, se bilaga 6.

Mätningarna utfördes för tre GlusterFS klientservrar:

- **OSF (skalbarhet- och prestandamätning)**
  I detta test utfördes prestanda- och skalbarhetsmätning i Oakland (se figur 20).

- **WVN (skalbarhet- och prestandamätning)**
  I detta test utfördes prestanda- och skalbarhetsmätning i Morgantown (se figur 20).

- **WSU (skalbarhet- och prestandamätning)**
  I detta test utfördes prestanda- och skalbarhetsmätning i Detroit (se figur 20).



Figur 20: Figuren visar mätningar av skalbarhet och prestanda då alla tre klientservrarna kördes samtidigt (se blå kurva) och var för sig (se röd, grå och gul kurva). På x-axeln representeras datahastigheten och på y-axeln representeras antalet användare som är uppkopplade till servrarna. *Bildkälla:* [103], se sidan 34.

I detta test, utfördes skalbarhet- och prestandamätning som en funktion av parallella samt separata servrar. Resultaten som erhölls i detta experiment gav en tydlig översikt på hur GlusterFS fungerade i en konkurrent och samverkande miljö. Det som kan observeras ur figuren är att totala prestandan var mycket högre när fler användare var uppkopplade jämfört med varje klientserver för sig. Detta är ett klart tecken för skalbarhet, vilket märks i den gråa kurvan då ökande antal användare inte försämrade prestandan trots att den uppmätta datahastigheten inte var så hög. Dock visade den röda och den gula kurvan sämre skalbarhet upp till ca 13 användare och det berodde på olika faktorer som exempelvis nätverkshastighet.

Kommentar

Testet visade för det objektbaserade lagringssystemet GlusterFS att skalbarheten för enstaka klientservrar var uppenbart bättre än när de kördes samtidigt, vilket tydligt framgår av figur 20. Dock var prestandan för var och en av klientservrarna avsevärt sämre än den totala uppmätta prestandan.

3.3.2 Disaster Recovery

Följande avsnitt beskriver de mätningar som gjordes gällande dataskydd i form av säkerhetskopiering och återställning.

3.3.2.1 Mätning av säkerhetskopiering i Ceph och GlusterFS

I arbetet ”Analysis of Six Distributed File Systems” genomfördes mätningar gällande säkerhetskopiering i de distribuerade filsystemen bland annat GlusterFS och Ceph. I detta examensarbete togs hänsyn endast till mätningar som beskrev hur effektiva dessa distribuerade filsystem var vid krasch av en lagringsenhet. Testerna mätte dataskydd i form av hur mycket MB av en datamängd som kan säkerhetskopieras.

För att utföra mätningarna, användes en plattform grid5000[6] som gav möjlighet att reservera samt skapa en miljö på ett par noder i olika kluster. Därefter så installerades de distribuerade filsystemen Ceph och GlusterFS och konfigurerades för att sedan kunna köras [40].

Testerna som utfördes utvärderade hur feltolerant ett system är när en av lagringsenheterna kraschade. Systemen kollade ifall den kraschade noden gick att spåra, om replikeringen fungerade effektivt samt om data fanns tillgänglig. Följande tester utfördes i infrastrukturen med konfiguration av varje DFS:

- **Ceph**
  I detta DFS användes 3 OSD (servrar) och 2 repliker.

- **GlusterFS**
  I detta DFS användes 4 servrar och 2 repliker.

[6] **Grid5000** är en testmiljö för olika datamätningar [63].

Ceph

Första mätningen utfördes på Ceph. Datamängden i storleksordning av 34 MB replikerades 2 gånger och fördelades mellan servrarna. Tabellen nedan visade den datamängden som tog upp lagringsutrymme före och efter tillägg av data.

Tabell 12: Tabellen visar den datamängden som tog upp lagringsutrymme före och efter tillägg av data på Ceph (se sidan 32) [40].

		Server1	**Server2**	**Server3**
Diskutrymme före		203 MB	204 MB	204 MB
Inlagd data med 2 repliker	34 MB			
Modifikation	73 MB	15 MB	34 MB	24 MB
Diskutrymme efter		218 MB	238 MB	228 MB

En lagringsenhet togs bort från klustret. Denna förändring upptäcktes av systemet som en krasch. Den datamängden som fanns i server1, kunde återhämtas och fördelas mellan server2 och server3. Tabellen nedan visade den datamängden som tog upp lagringsutrymme före och efter en krasch på en lagringsenhet.

Tabell 13: Tabellen visar den datamängden som tog upp lagringsutrymme före och efter en krasch på en lagringsenhet i Ceph. Minustecknet på modifikationen hos server1 kännetecknar att data flyttades från server1 och fördelades på de övriga servrarna (se sidan 33) [40].

	Server1	**Server2**	**Server3**
Diskutrymme före krasch	218 MB	238 MB	228 MB
Status	Avbrott	OK	OK
Modifikation	-15 MB	4 MB	11 MB
Diskutrymme efter krasch	203 MB	242 MB	239 MB

Ett test utfördes för att kontrollera om data var tillgänglig, vilket visade sig vara det. Den kraschade lagringsenheten startades om och ytterligare ett test utfördes för att kontrollera om alla servrarna var aktiva, vilket visade sig stämma.

GlusterFS

Andra mätningen utfördes på GlusterFS. Datamängden i storleksordning av 34 MB replikerades 2 gånger och fördelades mellan servrarna. Tabellen nedan visade den datamängden som tog upp lagringsutrymme före och efter tillägg av data.

Tabell 14: Tabellen visar den datamängden som tog upp lagringsutrymme före och efter tillägg av data på GlusterFS (se sidan 34) [40].

		Server1	**Server2**	**Server3**	**Server4**
Diskutrymme före		201 MB	201 MB	201 MB	201 MB
Inlagd data med 2 repliker	34 MB				
Modifikation		17 MB	17 MB	17 MB	17 MB
Diskutrymme efter	68 MB	218 MB	218 MB	218 MB	218 MB

En lagringsenhet togs bort från klustret. Denna förändring upptäcktes av systemet som en krasch. Den datamängden som fanns i server1, kunde inte återhämtas och fördelas mellan server2, server3 och server4. Tabellen nedan visade den datamängden som tog upp lagringsutrymme före och efter en krasch på en lagringsenhet.

Tabell 15: Tabellen visar den datamängden som tog upp lagringsutrymme före och efter en krasch på en lagringsenhet i Ceph. Minustecknet på modifikationen hos server1 kännetecknar att data flyttades från server1 och fördelades på de övriga servrarna (se sidan 34) [40].

	Server1	**Server2**	**Server3**	**Server4**
Diskutrymme före krasch	218 MB	218 MB	218 MB	218 MB
Status	Avbrott	OK	OK	OK
Modifikation	-17 MB	0 MB	0 MB	0 MB
Diskutrymme efter krasch	201 MB	218 MB	218 MB	218 MB

Ett test utfördes för att kontrollera om data var tillgänglig, vilket visade sig vara det. Den kraschade lagringsenheten startades om och ytterligare ett test utfördes för att kontrollera om alla servrarna var aktiva, vilket visade sig stämma.

Kommentar

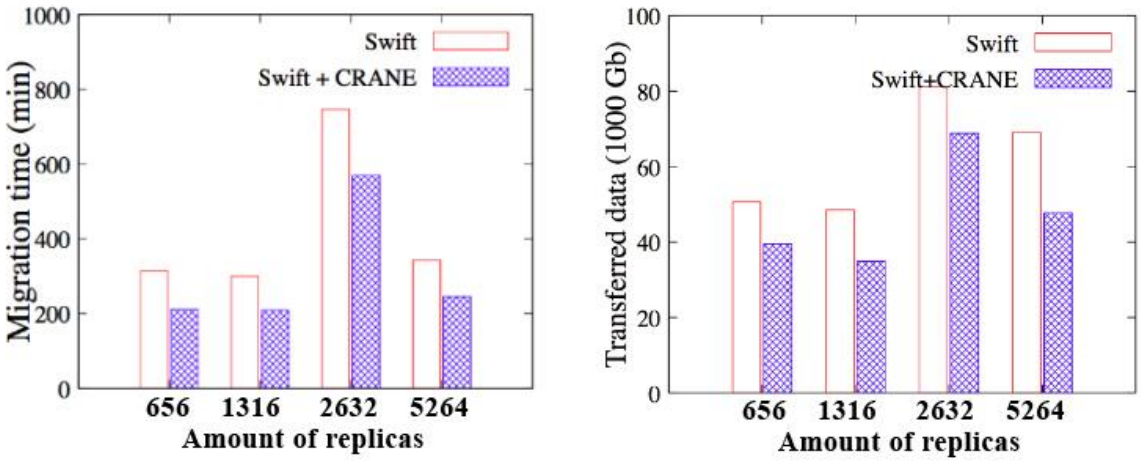
I ovanstående mätningar, observerades att den datamängden som existerade i den lagringsenheten som kraschade, kunde återhämtas och fördelas mellan övriga aktiva servrar då Ceph testades. Detta tydde på att replikering samt feltolerans kunde hanteras väl i denna DFS, vilket dock inte var fallet för GlusterFS då det data som existerade i den kraschade lagringsenheten inte kunde återhämtas och fördelas mellan övriga aktiva servrarna. Detta berodde på att antalet datarepliker i denna mätning var för lågt.

3.3.2.2 Mätning av säkerhetskopiering i SwiftStack

I arbetet ”On Optimizing Replica Migration in Distributed Cloud Storage Systems” genomfördes mätningar gällande säkerhetskopiering i det distribuerade filsystemet SwiftStack [104].

I detta examensarbete togs hänsyn endast till mätningar som beskrev hur effektivt det distribuerade filsystemet SwiftStack var vid säkerhetskopiering. Testen mätte dataskydd i form av replikering och överföring av icke-redundant data.

Infrastrukturen som mätningen utfördes på, bestod av fem lagringsservrar. Testen som utfördes utvärderade hur feltolerant SwiftStack var när en av lagringsservrarna kraschade. Mätningarna visade effektivitet vid överföring och replikering av datamängd. Under testerna, användes två varianter av SwiftStack; SwiftStack (standard) och SwiftStack med inbyggt CRANE algoritm[7]. Detta för att påvisa eventuella svagheter eller styrkor hos standardversionen av SwiftStack vid replikering och överföring. Figur 21 visar effektivitet vid replikering och överföring av icke-redundant data under serverkrasch.



Figur 21: Figuren visar mätningar av den totala tiden det tog för replikering och överföring av data (vänstra delen av figuren) samt mängden av data som replikerades och överfördes (högra delen av figuren) vid olika antal repliker. *Bildkälla:* [104], se sidan 7.

I vänstra delen av figur 21 visas den totala tiden det tog för replikering och överföring av data för samtliga fall då olika antal repliker överfördes. Det kan observeras ur denna del av figuren att SwiftStack behövde längre tid för att replikera och överföra data gentemot SwiftStack med inbyggd CRANE-algoritm.

[7] **CRANE** är en replikeringsalgoritm för distribuerade filsystem [104].

Anledning till att det tog längre tid för SwiftStack att replikera och överföra data var att SwiftStack med CRANE-algoritmen replikerade och överförde bara det data som verkligen behövdes, det vill säga ingen icke-redundant data varken replikerades eller överfördes mellan lagringsservrarna. Detta framgår av den högra delen av figuren.

Kommentar

Testet visade att CRANE-algoritmen effektiviserade SwiftStacks säkerhetskopiering gällande replikering och överföring av data då mindre data behövde överföras och replikeras. I ett system bestående av tusentals lagringsenheter kan en sådan svaghet försämra systemets prestanda.

3.3.2.3 Mätning av återställning i SwiftStack och GlusterFS

I arbetet "Performance Evaluation of Distributed Storage Systems" utfördes återställningsmätningar på SwiftStack och GlusterFS [64]. Systemet som mätningarna utfördes på, bestod av 24 lagringsnoder med ett totalt lagringsutrymme på 768 TB. Testerna mätte dataskydd i form av hur lång tid det skulle ta för ett distribuerat filsystem att återställas efter en katastrof.

I detta arbete genomfördes två återställningstester för samtliga DFS. Detta för att ta reda på hur lång tid det skulle ta att återuppta respektive system efter att en krasch har inträffat. Tabell 16 visar mätresultaten för återställningstiden av samtliga DFS.

Tabell 16: Tabellen visar återställningstiden för två återställningstester mellan SwiftStack och GlusterFS (se sidan 20) [64].

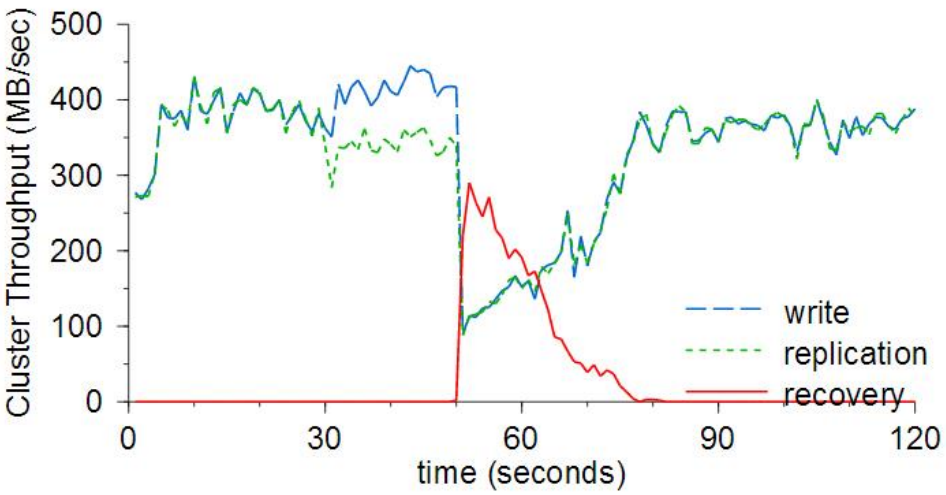
DFS	Återställningstest 1	Återställningstest 2
SwiftStack	9 hours 27 minutes (34020 s)	10 hours 16 minutes (36960 s)
GlusterFS	18 hours 27 minutes (66420 s)	18 hours 29 minutes (66540 s)

Kommentar

Det framgår av tabell 16 att SwiftStacks återställningstid var hälften så kort som GlusterFSs återställningstid i första testet. I andra återställningstestet tog det SwiftStack 1 timme mer att återställas, medan GlusterFS behöll ungefär samma återställningstid.

3.3.2.4 Mätning av återställning i Ceph

I Sage Weils doktorsavhandling "CEPH: RELIABLE, SCALABLE, AND HIGH-PERFORMANCE DISTRIBUTED STORAGE" utfördes bland annat mätningar på prestanda, skalbarhet och återställning [19]. I detta examensarbete togs hänsyn endast till mätningarna gällande återställning. Systemet som mätningen utfördes på, bestod av 20 lagringsenheter. Testerna mätte dataskydd i form av hur lång tid det skulle ta för ett distribuerat filsystem att återställas efter en katastrof. Figur 22 visar återställningstid efter att systemets två lagringsenheter har kraschat.



Figur 22: Figuren visade mätningen av återställningstiden (se den röda kurvan) för två lagringsenheter. På x-axeln representeras datahastigheten och på y-axeln representeras tiden. *Bildkälla:* [19], se sidan 170.

Kommentar

I figur 22 kan observeras att vid tiden 50 sekunder, kraschade två lagringsenheter och en återställning trädde i kraft. Återställningen pågick i dryga 30 sekunder och därefter återkom systemet till sitt ursprungliga läge. Det är viktigt att påpeka att under återställningstiden påverkades systemets prestanda men vid ungefär 51:a sekunden började prestandan att öka igen och uppnådde sin gamla nivå redan vid 80:e sekunden. Det är viktigt att nämna att i doktorsavhandlingen påpekas det av författaren att denna återställningstid som uppmättes är överdrivet lång samt att datahastigheten sjunker ovanligt mycket. Detta påstående motiveras genom att då mätningarna utfördes användes ett relativt litet system (ett fåtal lagringsenheter), medan i verkligheten där system som använder liknande lösningar är mycket större (tusentals lagringsenheter och uppåt) och sprider ut sina kopior till många fler lagringsenheter, därmed blir replikeringstiden kortare [19].

3.3.3 Läsning och skrivning

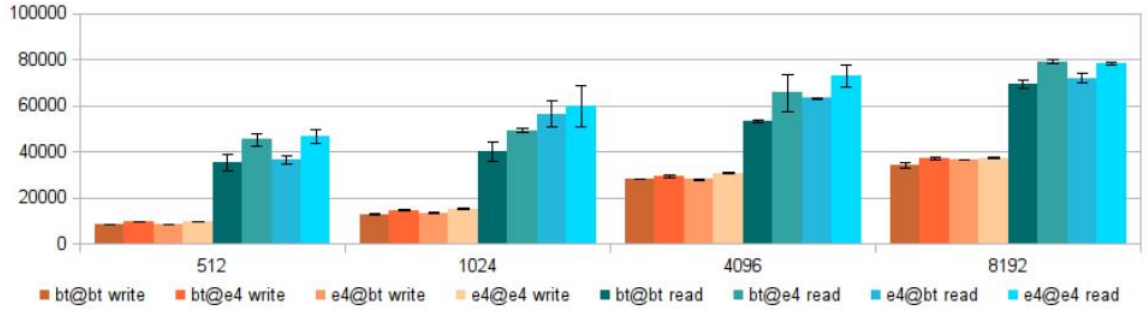
Följande avsnitt beskriver de mätningar som gjordes gällande prestanda för läsning och skrivning samt prestandaisolering.

3.3.3.1 Mätningar av läs- och skrivtester för Ceph och GlusterFS

I masterarbetet “Cloud backend deployment at Faculty of informatics MU” utfördes prestandamätningar mellan Ceph och GlusterFS [65]. Fem fysiska noder användes i en testmiljö, varav en nod konfigurerades som master och resterande identiska noder konfigurerades som slave. Masternoden användes som en DFS klient. Testerna mätte prestanda i form av antalet läsningar och skrivningar (IOPS).

Då de flesta DFS använde ett underliggande filsystem på hårddisken, utfördes prestandamätningar på två filsystem, ext4[8] och btrfs[9] i Ceph och GlusterFS. Vidare skriver författaren att eftersom GlusterFS inte hade stöd för blockbaserad lagring vid den tidpunkten då mätningen gjordes, användes två olika filsystem varav ena filsystemet låg GlusterFS på och det andra användes för att utföra mätningarna på. Därmed simulerades en virtuell blockbaseradmiljö. Exempelvis ”bt@e4” innebar att själva DFS låg på btrfs och mätningarna utfördes på ext4. På så sätt utfördes mätningar för alla kombinationer, det vill säga ”bt@bt”, ”e4@e4” och ”e4@bt”. Detsamma gjordes även för Ceph, trots att detta DFS hade stöd för blockbaserad lagring. Detta mest för att mätningarna skulle utföras under samma omständigheter.

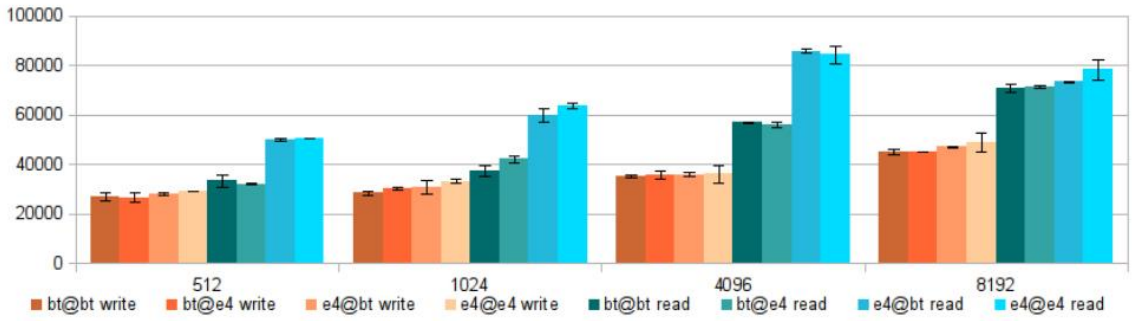
Första mätningen gjordes på GlusterFS av versionen 3.4-alpha. Följande graf (se figur 23) visade prestandamätningar för GlusterFS.



Figur 23: Figuren visar resultat efter att läs- och skrivtester gjordes för GlusterFS. *Bildkälla:* [65], se sidan 30.

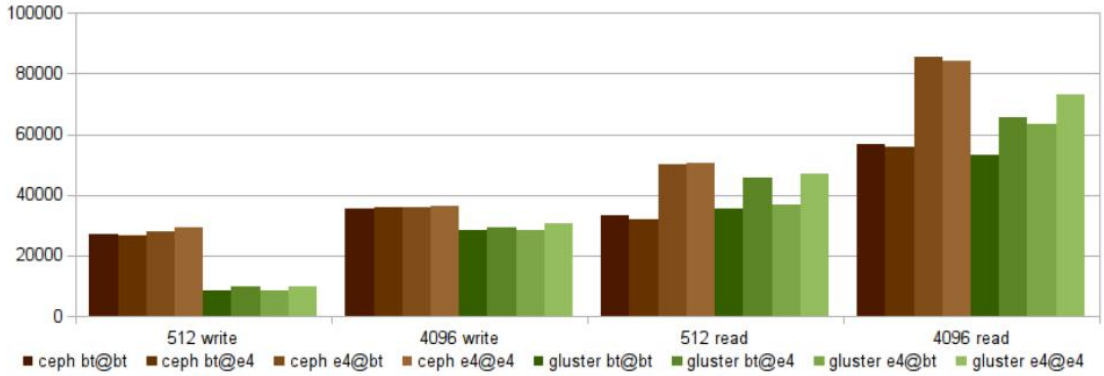
[8] **ext4** är en fjärde version av Linuxs filsystem [66].
[9] **Btrfs** är ett filsystem baserad på copy-on-write metoden, skapad av Oracle för användning i Linuxmiljöer [67].

Samma typ av mätning gjordes på Ceph av versionen 0.56. Följande graf (se figur 24) visade prestandamätningar för Ceph.



Figur 24: Figuren visar resultat efter att läs- och skrivtester gjordes för Ceph. *Bildkälla:* [65], se sidan 32.

Även en sammanlagd prestandajämförelse mellan Ceph och GlusterFS gjordes och presenterades i figur 25.



Figur 25: Figuren visar det sammanlagda mätresultatet mellan Ceph och GlusterFS. *Bildkälla:* [65], se sidan 34.

Kommentar

Det observerades i figur 25 att i de fallen då mätningarna utfördes på ext4, blev Cephs prestanda avsevärt bättre än GlusterFS gällande både skrivningar/läsningar vilka var av storleksordning av ett block[10]. Dock var GlusterFSs prestanda en aning bättre då mätningarna utfördes på btrfs vid läsning av block.

[10] **Block** är en sekvens av bytes. Standardstorlek av ett block är 4 KB [68, 69].

3.3.3.2 Mätningar av läs- och skrivtester för Ceph och GlusterFS i virtuella maskiner

I arbetet “Indexes for Distributed File/Storage Systems as a Large Scale Virtual Machine Disk Image Storage in Wide Area Network” utfördes prestandamätningar mellan Ceph, GlusterFS, Sheepdog och XtreemFS. I detta arbete lades stor vikt på prestandamätningar då parallella virtuella maskiner (VM) kördes. Mätningarna för Sheepdog och XtreemFS uteslöts i detta examensarbete då uppdragsgivarens önskemål var att Ceph, SwiftStack och GlusterFS i första hand skulle undersökas, vilket även framgick av avgränsningarna i kapitel 1. Vidare i samma arbete, användes 88 fysiska noder för prestandamätning som virtuella maskiner [70]. Testerna mätte prestanda i form av datahastighet (MB/s) samt prestandaisolering i form av hur väl resursfördelningen ser ut bland uppkopplade användare.

För att utvärdera effekten av ökande antal virtuella maskiner, genomfördes fyra olika typer av tester:

- **Parallel-1**
  I detta fall installerades en virtuell maskin åt gången och mätningarna gällande läsning och skrivning utfördes på 10 olika virtuella maskiner sekventiellt (se figur 26).

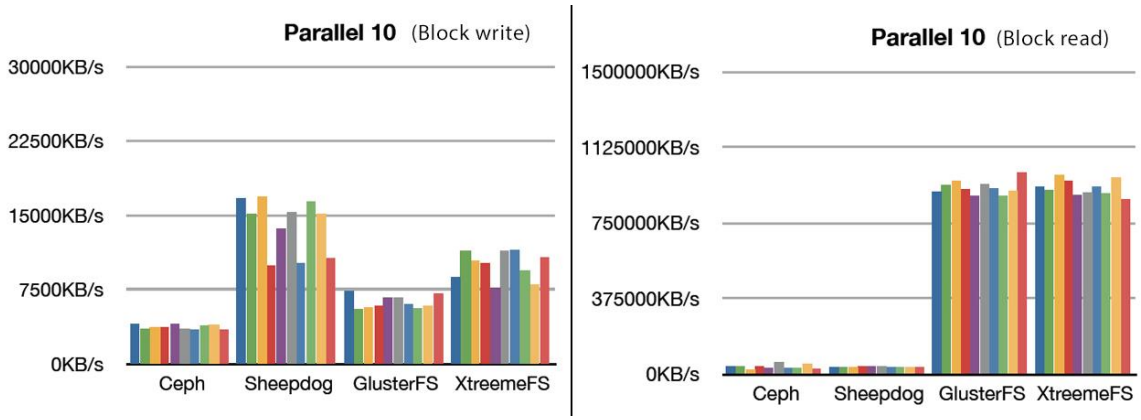
- **Parallel-10**
  I detta fall installerades 10 virtuella maskiner och mätningarna gällande läsning och skrivning utfördes parallellt (se figur 27).

- **Parallel-20**
  I detta fall installerades 20 virtuella maskiner och mätningarna gällande läsning och skrivning utfördes parallellt (se figur 28).

- **Parallel-44**
  I detta fall installerades 44 virtuella maskiner och mätningarna gällande läsning och skrivning utfördes parallellt (se figur 29).

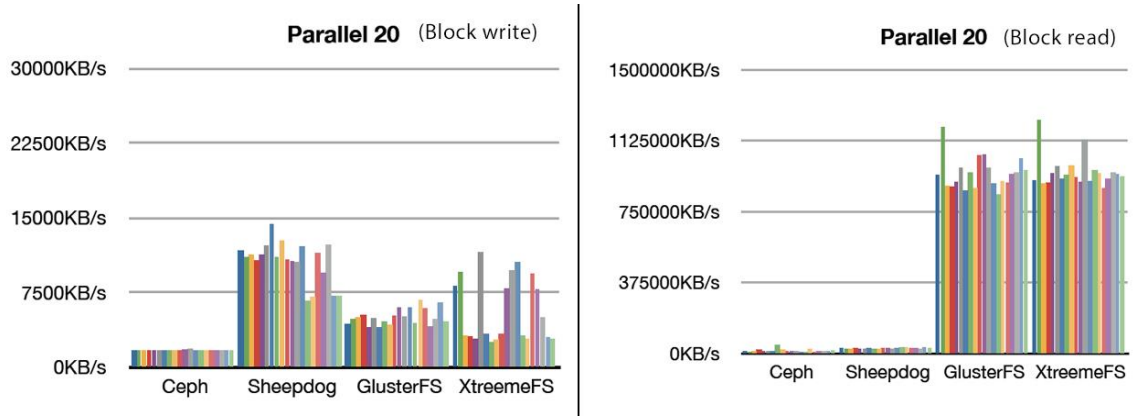
Figur 26: Figuren visar mätningar då 10 virtuella maskiner kördes sekventiellt vid mätning av skrivning och läsning. *Bildkälla:* [70], se sidan 5.

I figur 26 (block write) observerades att då 10 virtuella maskiner kördes sekventiellt då skrivtester utfördes, hade Ceph överlägset bäst prestanda jämfört med GlusterFS. I samma figur (block read) visades att då lästester utfördes, har GlusterFS avsevärt bättre prestanda än Ceph.



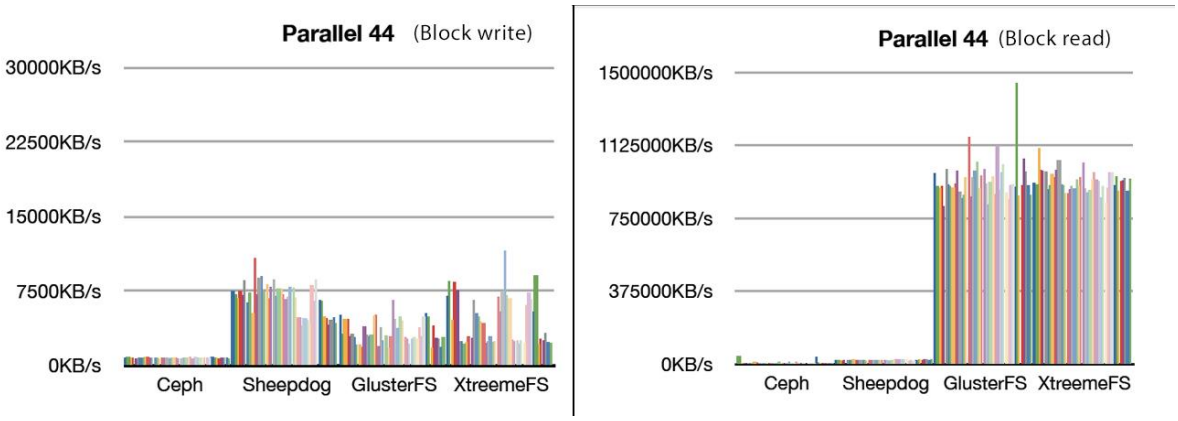
Figur 27: Figuren visar mätningar då 10 virtuella maskiner kördes parallellt vid mätning av skrivning och läsning. *Bildkälla:* [70], se sidan 5.

I figur 27 (block write) observerades att då skrivtesterna utfördes medan 10 virtuella maskiner kördes parallellt, hade GlusterFS överlägset bäst prestanda jämfört med Ceph. Dock visade testerna att Ceph hade mindre variationer då det gällde VM resursfördelning, vilket kunde läsas av staplarnas nivåer. I samma figur (block read) visades även att då lästester utfördes hade GlusterFS avsevärt bättre prestanda än Ceph.



Figur 28: Figuren visar mätningar då 20 virtuella maskiner kördes parallellt vid mätning av skrivning och läsning. *Bildkälla:* [70], se sidan 5.

I figur 28 (block write) observerades att då skrivtesterna utfördes medan 20 virtuella maskiner kördes parallellt, hade GlusterFS överlägset bäst prestanda jämfört med Ceph. Dock visade testerna att Ceph hade omärkligt små variationer, vilket kunde läsas av staplarnas nivåer. I samma figur (block read) visades även att då lästester utfördes hade GlusterFS avsevärt bättre prestanda än Ceph. Dock var variationerna gällande VM resursfördelning på GlusterFS markant frekventa.



Figur 29: Figuren visar mätningar då 44 virtuella maskiner kördes parallellt vid mätning av skrivning och läsning. *Bildkälla:* [70], se sidan 5.

I figur 29 (block write) observerades att då skrivtesterna utfördes medan 44 virtuella maskiner kördes parallellt, hade GlusterFS överlägset bäst prestanda jämfört med Ceph. Ceph hade ännu en gång bättre VM resursfördelning, vilket lästes av staplarnas nivåer. I samma figur (block read) visades även att då lästesterna utfördes hade GlusterFS avsevärt bättre prestanda än Ceph. Ännu en gång var variationerna gällande VM resursfördelning på GlusterFS markant frekventa.

Kommentar

Testerna visade att då mätningarna utfördes sekventiellt blev Ceph överlägset bäst. Dock sjönk Cephs prestanda ju fler VM som installerades och kördes parallellt, vilket framgick av figurerna 27, 28 och 29. Trots att Cephs prestanda sjönk ju fler VM som kördes parallellt, visade sig att Ceph hade bättre resursfördelning mellan VM vilket är en viktig egenskap när det gäller Virtual Machine Storage.

3.3.3.3 Mätningar av läs- och skrivtester för SwiftStack och GlusterFS

I arbetet "Performance Evaluation of Distributed Storage Systems" utfördes prestandamätningar gällande läs- och skrivning av data på SwiftStack och GlusterFS [64]. I detta examensarbete togs hänsyn till mätningar kring läs- och skrivprestanda samt CPU utnyttjandegrad. Systemet som mätningarna utfördes på, bestod av 24 lagringsnoder med ett totalt lagringsutrymme på 768 TB. Testerna mätte prestanda i form av datahastighet (MB/s) samt CPU utnyttjandegraden inom lagringsnoder och proxynoder.

I detta arbete genomfördes fyra olika tester gällande läs- och skrivprestanda:

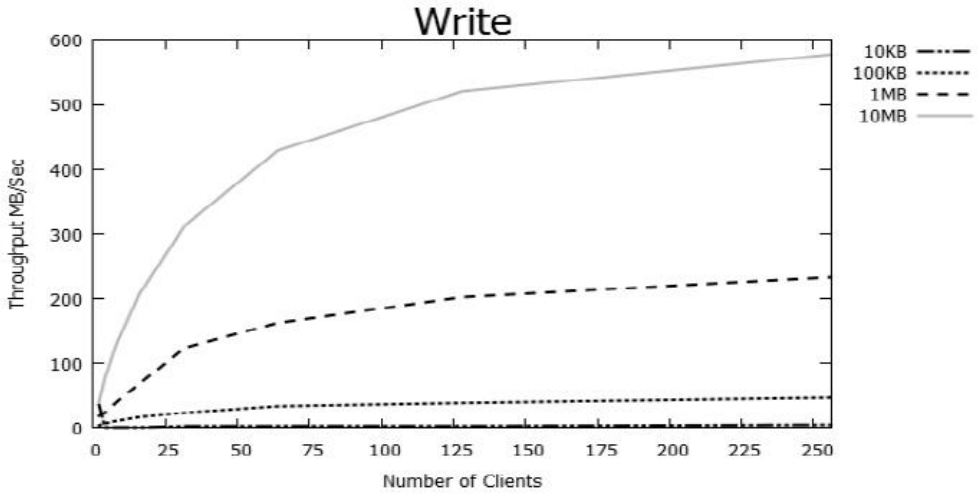
- **10 KB**
  I detta fall utfördes läs- och skrivtester för filer i storleksordningen 10 KB.

- **100 KB**
  I detta fall utfördes läs- och skrivtester för filer i storleksordningen 100 KB.

- **1 MB**
  I detta fall utfördes läs- och skrivtester för filer i storleksordningen 1 MB.

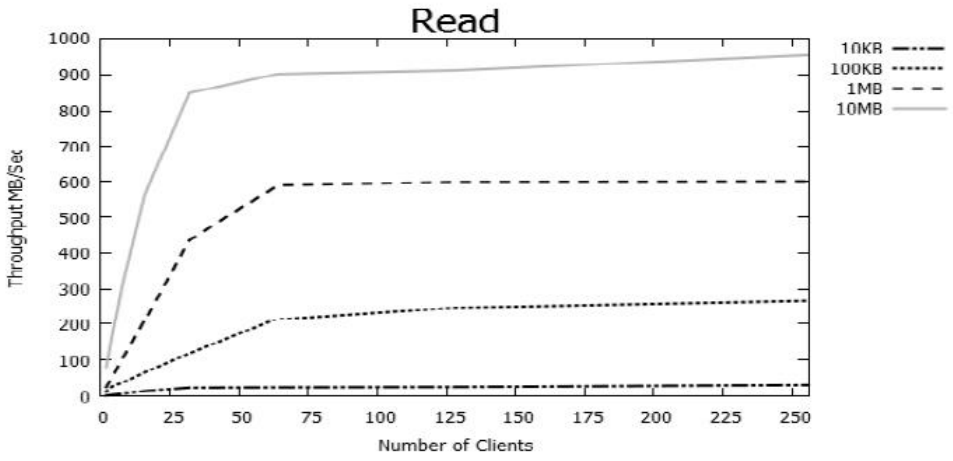
- **10 MB**
  I detta fall utfördes läs- och skrivtester för filer i storleksordningen 10 MB.

I kommande undersökning uteslöts analys och jämförelse av läs- och skrivtester för 10 KB filer då dessa mer eller mindre visade lika prestanda på alla mätningar. Första mätningen utfördes på SwiftStack, där skrivtesterna visas i figur 30. Ur figuren kan observeras att SwiftStacks prestanda blev bättre ju större filer som skrevs. En annan egenskap som SwiftStack visade vid skrivning av större filer är att prestandan ökade trots att fler användare kopplade upp sig mot systemet, detta visas av kurvan som representerar filstorleken på 10 MB. Vid skrivning av mindre filer började kurvorna att plana ut sig fortare ju fler användare som kopplade upp sig mot systemet. Detta visas av kurvorna som representerar filstorlekar på 10 KB och 100 KB.



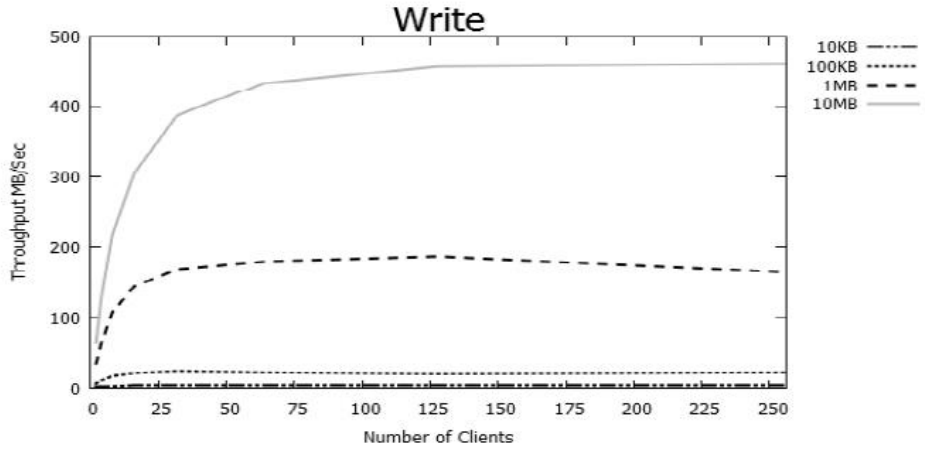
Figur 30: Figuren visar skrivtester för SwiftStack för samtliga filstorlekar.
Bildkälla: [64], se sidan 12.

Andra mätningen som utfördes på SwiftStack, omfattade lästesterna. Detta visas i figur 31. Ur figuren kan observeras att SwiftStacks prestanda blev avsevärt bättre ju större filer som lästes. En annan karakteristik som visas i figuren var att vid läsning av större filer, ökade prestandan trots att fler användare kopplade upp sig till systemet, se kurvan som representerar filstorleken på 10 MB. Dock började kurvan bli jämnare redan vid 60 användare, vilket tyder på att SwiftStack hade en bra prestandaisolering.



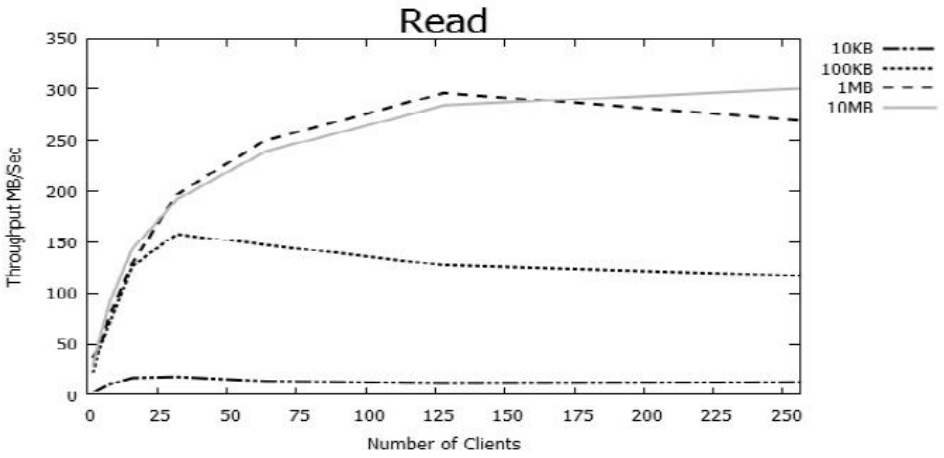
Figur 31: Figuren visar lästester för SwiftStack för samtliga filstorlekar.
Bildkälla: [64], se sidan 13.

Den tredje mätningen utfördes på GlusterFS, där skrivtesterna visas i figur 32. Ur figuren kan observeras att GlusterFSs prestanda blev bättre ju större filer som skrevs. Det kan även observeras att prestandan hade en brantare kurva upp till ungefär 30 användare vid skrivning av filer i storleksordningen 10 MB, därefter började kurvan plana ut. Vid ungefär 125 användare blev datahastigheten mer jämnfördelad och behöll samma kontur upp till 250 användare, vilket var maximalt antal användare som användes i samtliga mätningar. Däremot vid läsning av 1 MB filer, visade prestandan en växande tendens upp till 125 användare. Dock blev inte datahastigheten lika hög som vid läsning av 10 MB filer, utan kurvan började avta när fler än 125 användare anslöt sig till systemet.



Figur 32: Figuren visar skrivtester för GlusterFS för samtliga filstorlekar.
Bildkälla: [64], se sidan 13.

Fjärde mätningen som utfördes på GlusterFS, omfattade lästesterna. Detta visas i figur 33. Ur figuren kan observeras att GlusterFS prestanda blev bättre ju större filer som lästes. Dock visades inga större prestandaskillnader vid läsning av 1- och 10 MB filer upp till ungefär 125 användare. Därefter började kurvan för 1 MB filer att avta medan kurvan för 10 MB filer fortsatte att växa långsamt.



Figur 33: Figuren visar lästester för GlusterFS för samtliga filstorlekar.
Bildkälla: [64], se sidan 14.

För samtliga mätningar gällande läs- och skrivtester, utfördes tester för CPU utnyttjandegraden. Nedanstående tabeller (se tabell 17, 18, 19 och 20) visar mätningar för CPU utnyttjandegraden för läs- och skrivtesterna. I tabellen står S CPU för ”storage nodes CPU” och P CPU för ”proxy node CPU”.

Tabell 17: Tabellen visar mätresultaten för skrivning av filer samt CPU utnyttjandegrad på lagringsnoder och proxynoder för SwiftStack (se sidan 27) [64].

Write	100 KB			1 MB			10 MB		
Number of clients	MB/s	S CPU %	P CPU %	MB/s	S CPU %	P CPU %	MB/s	S CPU %	P CPU %
2	3	30	4	18	49	12	40	29	6
4	6	33	7	23	49	12	76	29	19
8	10	35	11	39	53	12	129	31	22
16	16	46	16	68	80	26	207	56	28
32	23	49	23	124	87	27	314	62	41
64	33	55	23	162	85	25	429	87	74
128	38	86	23	202	84	47	521	95	81
256	46	76	44	233	92	60	578	96	92

De erhållna resultaten i tabell 17 visar att S CPU utnyttjandegraden ökade ju fler filer som skrevs och ju fler användare som anslöt sig till systemet. Dock var utgångsutnyttjandegraden någorlunda högre då 1 MB filer lästes. Detsamma gällde även för P CPU utnyttjandegraden.

Tabell 18: Tabellen visar mätresultaten för läsning av filer samt CPU utnyttjandegrad på lagringsnoder och proxynoder för SwiftStack (se sidan 27) [64].

Read	100 KB			1 MB			10 MB		
Number of clients	MB/s	S CPU %	P CPU %	MB/s	S CPU %	P CPU %	MB/s	S CPU %	P CPU %
2	12	30	4	25	48	6	80	22	6
4	26	30	10	51	48	11	164	24	14
8	38	33	17	106	51	17	316	29	30
16	66	38	35	212	62	38	565	35	55
32	118	41	53	435	65	70	849	38	78
64	215	53	82	589	67	90	901	48	93
128	248	64	98	598	69	80	911	51	98
256	267	69	99	600	86	100	953	55	99

De erhållna resultaten i tabell 18 visar att S CPU utnyttjandegraden ökade ju fler filer som lästes och ju fler användare som anslöt sig till systemet. Dock ökade P CPU utnyttjandegraden avsevärt när fler än 32 användare anslöt sig till systemet för samtliga filstorlekar.

Tabell 19: Tabellen visar mätresultaten för skrivning av filer samt CPU utnyttjandegrad på lagringsnoder och proxynoder för GlusterFS (se sidan 28) [64].

Write	100 KB			1 MB			10 MB		
Number of clients	MB/s	S CPU %	P CPU %	MB/s	S CPU %	P CPU %	MB/s	S CPU %	P CPU %
2	6	6	9	34	6	9	63	5	10
4	11	11	18	64	13	19	130	12	20
8	17	18	35	107	18	42	217	24	45
16	21	19	46	144	22	44	304	30	52
32	24	24	38	168	24	48	387	42	56
64	22	34	32	179	24	50	434	43	59
128	20	22	34	186	35	51	458	41	69
256	22	37	39	164	31	44	462	41	67

De erhållna resultaten i tabell 19 visar att utnyttjandegraden för S CPU och P CPU låg ungefär på samma nivå. Dock var utgångsutnyttjandegraden för P CPU lite högre än för S CPU för samtliga filstorlekar. Det kan också observeras att utnyttjandegraden började avta efter att fler än 128 användare anslöt sig till systemet då filer av storleksordningarna 1 MB och 10 MB skrevs.

Tabell 20: Tabellen visar mätresultaten för läsning av filer samt CPU utnyttjandegrad på lagringsnoder och proxynoder för GlusterFS (se sidan 29) [64].

Read	100 KB			1 MB			10 MB		
Number of clients	MB/s	S CPU %	P CPU %	MB/s	S CPU %	P CPU %	MB/s	S CPU %	P CPU %
2	23	4	10	38	6	11	30	6	9
4	44	8	11	45	8	18	53	12	20
8	72	15	49	79	12	43	92	14	38
16	126	24	68	129	24	58	143	18	46
32	158	32	72	197	28	71	192	22	53
64	147	35	80	250	36	77	239	24	55
128	127	30	83	296	27	78	284	28	64
256	117	45	62	270	40	80	300	32	66

De erhållna resultaten i tabell 20 visar att utnyttjandegraden för S CPU och P CPU låg ungefär på samma nivå. Dock var utgångsutnyttjandegraden för P CPU lite högre än för S CPU för samtliga filstorlekar. Det kan även observeras att P CPU utnyttjandegraden för läsning av 1 MB filer var betydligt högre då 256 användare anslöt sig till systemet jämfört med läsning av 100 KB och 10 MB filer för samma antal användare.

Kommentar

Skrivtesterna för både SwiftStack och GlusterFS visade att båda DFS har en ökad prestanda ju fler användare som anslöt sig till systemet samt ju större filer som skrevs. En uppenbar skillnad som kan observeras är att vid skrivning av 10 MB filer, steg datahastighetskurvan mycket brantare för GlusterFS medan SwiftStacks datahastighetskurva hade en mer logaritmisk form. Dock uppnådde SwiftStack en högre datahastighet och fortsatte växa även vid 250 användare, medan GlusterFS uppnådde sitt maximum vid ungefär 125 användare och bibehöll samma nivå upp till 250 användare. Detta kunde innebära att GlusterFS hade en förmåga att jämnare fördela resurser då många användare var uppkopplade samtidigt. Bland andra resurser påverkades CPU utnyttjandegraden, vilket framgår av tabellerna för skrivning (se tabell 17 och 19), där det kan observeras att skillnaderna i CPU utnyttjandegraden mellan dessa DFS. När det gällde lästester, visade SwiftStack att datahastigheten kom upp till 850 MB/s redan vid 30 användare, därefter ökade datahastigheten långsammare och vid 75 användare bibehöll kurvan sin form ända till 250 användare. SwiftStack visade mycket bättre datahastighet när det gällde läsning.

Å andra sidan visade GlusterFS en sämre datahastighet för samtliga filstorlekar, jämfört med SwiftStack. Förutom den låga datahastigheten, var det även uppenbart att prestanda vid läsning av 1- och 10 MB filer var ungefär samma upp till 125 användare (se figur 33), vilket inte var fallet för SwiftStack där skillnaderna mellan läsning av 1- och 10 MB filer var uppenbara (se figur 31). När det gällde CPU utnyttjandegraden visade SwiftStack en mycket högre utnyttjandegrad av CPU (se tabell 18), detta kunde bero på den höga datahastigheten som SwiftStack kom upp till. GlusterFS behöll en jämnare CPU utnyttjandegrad även för läsning av filer (se tabell 20).

3.4 Övriga egenskaper hos öppna distribuerade filsystem

Övriga egenskaper (exempelvis kostnader) som inte gick att mäta, presenteras i detta avsnitt för öppna distribuerade filsystem utifrån faktainsamling. Tabell 21 lyfter fram de egenskaperna som samtliga öppna distribuerade filsystem har.

Tabell 21: Tabellen visar de övriga egenskaperna som GlusterFS och Ceph har.

Övriga egenskaper	GlusterFS	Ceph	SwiftStack
Thin provisioning	Nej	Ja [71]	Nej
Data deduplicering	Nej	Nej	Nej
Commodity Hardware	Ja [72]	Ja [73]	Ja [74]
Datatillgänglighet	RAID 1 [40]	Replikering [40]	Replikering [75]
Administration/underhåll	Linuxutvecklare [76]	Linuxutvecklare [77]	Linuxutvecklare [78]
Vendor lock-in[11]	Nej	Nej	Nej
Snapshotting	Ja [79]	Ja [80]	Nej

[11] **Vendor lock-in** innebär att ett system är låst till endast en tillverkares komponenter [81].

3.5 Kostnadsuppskattning för distribuerade filsystem

Följande avsnitt presenterar sammanställning av kostnader för både kommersiella- och öppna distribuerade filsystem under en fem års period. Fem års period valdes då hårdvaran som skulle kunna användas för uppsättning av det öppna DFS slits ut och tappar i värdet samt att ny teknologi introduceras ungefär vart femte år. Därmed borde hårdvaran bytas ut mot en ny.

Sammanställning av kostnader för öppna DFS, baserades på kostnader för utrustning, elförbrukning och administration som krävs för att sätta upp och underhålla ett DFS medan kostnader för kommersiella tjänster baserades på de prisplaner som de kommersiella tjänsterna listar på sina hemsidor. Systemutrustningen som kostnadsuppskattningen baserades på, valdes just för att det företaget som examensarbetet utfördes på använder den typen av utrustning. Systemet är uppbyggt av två typer av lagring, en snabblagring och en normallagring. Snabblagring består av tre servrar varav varje server har två HDD för installation av OS, fem SSD för caching och fem HDD för lagring. Detta ger en total effektiv lagring på 10 TB (5 HDD á 2 TB) på ena servern, då resten av det tillgängliga lagringsutrymmet på de två andra servrarna används för kopior av data. Normallagring består av tre servrar, varav varje server innehåller två HDD för installation av OS och 15 HDD för lagring. Alltså, existerar det inga SSD-diskar på normallagring. Det effektiva lagringsutrymmet på den normala lagringen blir 30 TB (15 HDD á 2 TB). Administrationskostnader baserades på en snittlön för en Linuxadministratör, då samtliga öppna distribuerade filsystem kan administreras av en Linuxadministratör. Det är viktigt att påpeka att en dag per vecka räcker för en Linuxadministratör att betjäna samtliga servrar. Detta konstaterande kommer att påverka den totala kostnaden för administration då endast en femtedel av den totala lönesumman ska räknas in i kostnadskalkylen.

Den sammanlagda kostnaden för öppna distribuerade filsystemet presenteras nedan. För mer detaljerad information kring initialkostnaden, se bilaga 7. Det bör påpekas att hårddiskarna som ingick i kostnadskalkylen har en livslängd mellan 3 till 5 år. I examensarbetet utgicks det ifrån att dessa hårddiskar håller i 3 år, därefter byts de ut mot nya hårddiskar vilket innebär en extra kostnad som tillkommer vart tredje år. Det slutgiltiga priset per GB/månad kommer att grundas på den totala kostnaden och 40 TB lagringsutrymmet (10 TB snabblagring + 30 TB normallagring).

Total initialkostnad: 925 025 SEK.

Hårddiskbyteskostnad: Diskarna förväntas bytas ut en gång under en 5 års period, vilket skapar ytterligare en kostnad på 161 853 SEK.

Administrationskostnader: 32 000 SEK/månad [82]. Social och arbetsgivaravgifter ökar de totala administrationskostnader till 42 054 SEK/månad (504 648 SEK på en 5 års period) [105].

Elförbrukningskostnader: Nedanstående tabell beräknar den totala elförbrukningen inom ett årsperiod (se tabell 22).

Tabell 22: Tabellen visar elförbrukningen för varje utrustning samt den totala elförbrukningen. Förbrukningen (kWh) beräknas genom: effekt * utnyttjandegrad * antal.

Utrustning	**Effekt (kWh)**	**Utnyttjandegrad (%)**	**Antal (st)**	**Förbrukning (kWh)**	**Förbrukning (kWh/år)**
Server	1,4	50	6	4,2	36792
Switch	0,6	50	2	0,6	5256
Total förbrukning (kWh/år)					42048

Elpriset som användes i beräkningen, hämtades från EON och baseras på det totala antalet kWh/år samt postnumret där företaget finns [83]. För kalkyl av elpriset, se bilaga 8.

Den totala elförbrukningskostnaden beräknades till: 0,6748 * 42048 * 5 ≈ 141870 SEK.

Total kostnad (öppna DFS): Nedanstående tabell summerar den totala kostnaden för fem års period (se tabell 23).

Tabell 23: Tabellen visar summan för typ av kostnad samt den totala kostnaden för fem års period.

Typ av kostnad	**Initialkostnad**	**Hårddiskbyteskostnad**	**Administrationskostnader**	**Elförbrukningskostnader**
Kostnad (SEK)	925 025	161 853	504 648	141870
Total kostnad (SEK)	1 733 396			

Pris per GB/månad presenteras nedan (se tabell 24).

Tabell 24: Tabellen visar pris per GB/månad för ett öppet DFS.

Typ av lagring	Snabblagring	Normallagring
Lagringsutrymme (TB)	10	30
Totalkostnad/2 (SEK)	866 698	866 698
Pris per GB/månad (SEK)	1,44 ≈ 1,4	0,4814 ≈ 0,48

Nedanstående tabell summerar kostnader per GB/månad för både kommersiella- och öppna DFS (se tabell 25). Valutakursen som användes för omvandling av priset för samtliga kommersiella datalagringslösningar från amerikanska dollar till svenska kronor, togs den 20 juli 2016. Priset för Microsofts och Googles normallagring erhölls som ett snitt mellan det högsta och det lägsta priset.

Tabell 25: Tabellen visar pris per GB/månad för kommersiella- och öppna DFS.

Företag	**Snabblagring**	**Pris per GB/månad för snabblagring (SEK)**	**Normallagring**	**Pris per GB/månad för normallagring (SEK)**
Google	Standard Storage – Local SSD	≈ 1,88 [84]	DRA Storage – Persisent Disks	≈ 0,75 [84]
Amazon	Standard Storage – General Purpose (Provisioned IOPS SSD)	≈ 1,29 [85]	Standard Storage – Elastic Block Storage (Throughput Optimized HDD)	≈ 0,46 [85]
Microsoft	Premium Disks	≈ 1,33 [86]	LRS, GRS, RA-GRS – Standard Disks	≈ 0,60 [86]
Öppet DFS	Snabb (SSD-lösning)	1,4	Normallösning	0,48

3.6 Summering av resultaten

Följande avsnitt sammanfattar och summerar resultaten som åstadkommits vid undersökning av kommersiella- och öppna distribuerade filsystem. Detta avsnitt sammanställer undersökta mätningar, egenskaper och övriga parametrar för samtliga kommersiella- och öppna distribuerade filsystem och visas i olika tabeller.

3.6.1 Summering av resultaten för kommersiella distribuerade filsystem

Nedanstående tabeller (se tabell 26 och 27) summerar resultaten som åstadkommits vid undersökning av kommersiella distribuerade filsystem. Notera att i rubrik 3.1.1.1 och 3.1.2.1 undersöktes olika typer av respektive lösning. Exempelvis för Google undersöktes tre olika typer av datalagringslösnimngar; Standard Storage, DRA Storage och Nearline Storage. Dock i rubrik 3.1.1.2 och 3.1.2.2 framgick det inte av det undersökta arbetet vilken typ av respektive datalagringslösning som behandlades utan Google, Amazon och Microsoft undersöktes som helhet och inte typvis som i föregående arbete.

Tabell 26: Tabellen visar en summering av alla mätningar som undersökts för kommersiella DFS.

Rubrik	**Typ av mätning**	**Summering av resultatet**
3.1.1.1 *Mätning av läs- och skrivtester mellan Google, Microsoft och Amazons datalagringslösningar*	Prestanda för läsning och skrivning (datahastighet).	Amazon Standard visade bäst läshastighet (7,12 MB/s) och Microsoft Azure Storage visade bäst skrivhastighet (6,71 MB/s).
3.1.1.2 *Mätning av läs- och skrivtester mellan Google, Microsoft och Amazon*	Prestanda för läsning och skrivning (datahastighet).	Prestanda mellan de olika datalagringslösningarna varierade inte betydligt mycket. Dock de nedladdningstesterna som utfördes på mätplatsen A visade avsevärda prestandaavvikelser, vilket kan bero på nätverkshastigheten.

Tabell 27: Tabellen visar en summering av alla jämförelser som undersökts för kommersiella DFS.

Rubrik	**Typ av jämförelse**	**Summering av resultatet**
3.1.2.1 *Jämförelse av kostnader mellan Googles, Microsofts och Amazons datalagringslösningar*	Kostnader (pris per GB/månad).	Microsofts datalagringslösning RA-GRS var det dyraste alternativet och Googles datalagringslösning Nearline Storage var det billigaste alternativet bland samtliga datalagringslösningar.

3.1.2.2 *Jämförelse av kostnader mellan Google, Microsoft och Amazon*	Kostnader (pris per GB/månad).	Tjänsteerbjudanden från Microsoft och Amazon var de dyraste alternativen och tjänsteerbjudandet från Google var det billigaste alternativet.

3.6.2 Summering av resultaten för öppna distribuerade filsystem

Nedanstående tabell (se tabell 28) summerar resultaten som åstadkommits vid undersökning av öppna distribuerade filsystem.

Tabell 28: Tabellen visar en summering av alla mätningar som undersökts för öppna DFS.

Rubrik	**Typ av mätning**	**Summering av resultatet**
3.3.1.1 *Mätning av dataarkivering i SwiftStack och Ceph*	Prestanda för läsning och skrivning (datahastighet) samt skalbarhet.	De två objektbaserade lagringssystem hade liknande prestanda då enstaka filer skrevs. Ceph uppnådde sin maxgräns vid skrivning av stora filer. Vid ökat antal parallella processer visade Ceph bättre prestanda än SwiftStack samt en optimalare datadistribution genom klustret.
3.3.1.2 *Mätning av dataarkivering i GlusterFS*	Prestanda för läsning och skrivning (datahastighet) samt skalbarhet.	Den uppmätta skalbarheten för det objektbaserade lagringssystemet GlusterFS för enstaka klientservrar var uppenbart bättre än när de kördes samtidigt. Dock var prestandan för var och en av klientservrarna avsevärt sämre än den totala uppmätta prestandan.
3.3.2.1 *Mätning av säkerhetskopiering i Ceph och GlusterFS*	Dataskydd (säkerhetskopiering).	Den datamängden som existerade i den lagringsenheten som kraschade, kunde återhämtas och fördelas mellan övriga aktiva servrar då Ceph testades. Detta var dock inte fallet för GlusterFS då det data som existerade i den kraschade lagringsenheten inte kunde återhämtas för att sedan fördelas mellan övriga aktiva servrarna. Detta berodde på att antalet datarepliker i denna mätning var för lågt.

3.3.2.2 *Mätning av säkerhetskopiering i SwiftStack*	Dataskydd (säkerhetskopiering).	CRANE-algoritmen effektiviserade SwiftStacks säkerhetskopiering gällande replikering och överföring av data då mindre data behövde överföras och replikeras. I ett system bestående av tusentals lagringsenheter kan en sådan svaghet försämra systemets prestanda.
3.3.2.3 *Mätning av återställning i SwiftStack och GlusterFS*	Dataskydd (återställning).	SwiftStacks återställningstid var hälften så kort som GlusterFSs återställningstid i första testet. I andra återställningstestet tog det SwiftStack 1 timme mer att återställas, medan GlusterFS behöll ungefär samma återställningstid.
3.3.2.4 *Mätning av återställning i Ceph*	Dataskydd (återställning).	Återställningen i det distribuerade filsystemet Ceph pågick i dryga 30 sekunder vid ungefär 50:e sekunden av mätningen. Systemets prestanda påverkades under återställningen men vid ungefär 51:a sekunden började prestandan att öka igen och uppnådde sin gamla nivå redan vid 80:e sekunden.
3.3.3.1 *Mätningar av läs- och skrivtester för Ceph och GlusterFS*	Prestanda för läsning och skrivning (IOPS).	De fallen då mätningarna utfördes på ext4, blev Cephs prestanda avsevärt bättre än GlusterFS gällande både skrivningar/läsningar. Dock var GlusterFSs prestanda en aning bättre då mätningarna utfördes på btrfs vid läsning av block.
3.3.3.2 *Mätningar av läs- och skrivtester för Ceph och GlusterFS i virtuella maskiner*	Prestanda för läsning och skrivning (datahastighet) samt prestandaisolering.	Då mätningarna utfördes sekventiellt blev Ceph överlägset bäst. Dock sjönk Cephs prestanda ju fler VM som installerades och kördes parallellt. Trots att Cephs prestanda sjönk ju fler VM som kördes parallellt, visade sig att Ceph hade bättre resursfördelning mellan VM.

3.3.3.3 *Mätningar av läs- och skrivtester för SwiftStack och GlusterFS*	Prestanda för läsning och skrivning (datahastighet) samt CPU utnyttjandegrad.	Både SwiftStack och GlusterFS visade en ökad prestanda ju fler användare som anslöt sig till systemet samt ju större filer som skrevs. GlusterFS hade en förmåga att jämnare fördela resurser då många användare var uppkopplade samtidigt. Det fanns även skillnader i CPU utnyttjandegraden mellan dessa DFS vid skrivning. Gällande läsning av filer, uppnådde SwiftStack betydligt bättre datahastighet. GlusterFS visade en sämre datahastighet för samtliga filstorlekar. CPU utnyttjandegraden för SwiftStack var väldigt hög, på grund av den höga datahastigheten som SwiftStack uppnådde vid läsning. Däremot behöll GlusterFS en jämnare CPU utnyttjandegrad.

4 Analys och diskussion

Målet med detta examensarbete var att undersöka olika kommersiella- och öppna distribuerade filsystem. Dock skulle ingen djupare undersökning utföras på kommersiella distribuerade filsystem (se 1.3 avgränsningar) och därmed användes undersökning av öppna distribuerade filsystem som en grund för att kunna bestämma en optimal öppet distribuerat filsystem som lämpar sig bäst för uppdragsgivarens önskemål. Undersökningen gick ut på att analysera användningsfallen *Cold Storage, High Performance Storage* och *Virtual Machine Storage* för respektive öppna distribuerade filsystem. Analys av resultaten av de undersökta arbeten visade både styrkor och svagheter hos samtliga distribuerade filsystem, beroende på vilken version av dessa distribuerade filsystem som testades vid respektive tillfälle.

I de arbeten som användes som underlag vid undersökning av användningsfallet *High Performance Storage* för samtliga öppna distribuerade filsystem, visade Ceph bäst prestandaegenskaper bland samtliga mätningar, detta framgår av avsnitt 3.3.3. Mätningar som presenterades i detta avsnitt visade att Ceph hade överlägset bäst både läs- och skrivprestanda jämfört med GlusterFS i två olika arbete. Vidare i avsnittet 3.3.1 utfördes mätningar för dataarkivering samt läs- och skrivmätningar. I detta arbete visade Ceph bäst både läs- och skrivprestanda även i jämförelse med SwiftStack.

I de arbeten som användes som underlag vid undersökning av användningsfallet *Virtual Machine Storage* för samtliga öppna distribuerade filsystem, visade Ceph bäst prestandaisolering bland samtliga mätningar. Ett av de analyserade arbetena (se avsnitt 3.3.1.1) visade att Ceph hade jämnare resursfördelning jämfört med SwiftStack. Det framgår tydligt av figurerna 18 och 19 att Ceph hade en jämnare resursfördelning ju fler användare som anslöt sig till systemet. Jämn resursfördelning är en parameter som kännetecknar prestandaisolering och därmed *Virtual Machine Storage*. Även i avsnitt 3.3.3.1 och 3.3.3.2 där mätningarna utfördes mellan Ceph och GlusterFS, visade Ceph jämnare resursfördelning vilket framgår av figurerna 24, 25, 26, 27, 28 och 29. Även när GlusterFS och SwiftStack jämfördes, visade dessa två distribuerade filsystem prestandaisolering, vilket framgår av samtliga mätningar i avsnitt 3.3.3.3 (se figur 30, 31, 32 och 33). Dock uppnås prestandaisolering då mindre filer som läses och skrivs samt ju fler användare som är uppkopplade till systemet. I de testerna utförda i avsnitt 3.3.3.3 visade GlusterFS bättre resursfördelning jämfört med SwiftStack vid högre antal användare. Detta tack vare Elastic Hash Algorithm som istället för att hålla reda på metadataindex, så allokerar denna data algoritmiskt, det vill säga den enda information den behöver veta är filens logiska väg och namn [89].

Det framgick av målsättningen att förutom *High Performance Storage* och *Virtual Machine Storage* skulle även undersökning av användningsfallet *Cold Storage* undersökas för samtliga öppna distribuerade filsystem.

Det visade sig i en utav mätningarna i avsnitt 3.3.1.1 att Ceph hade bättre skalbarhetsegenskaper jämfört med SwiftStack (se figur 19). Detta tack vare CRUSH-algoritmen vars unika replikeringsförmåga möjliggjorde att Ceph effektivt distribuerade olika objekt genom lagringssystemet. SwiftStacks förmåga att hantera parallella processer visade sig vara sämre än Cephs, eftersom att enligt arkitekturen som SwiftStack bygger på, måste alla dessa processer att gå igenom en proxynod, detta framgick av resultatet i avsnitt 3.3.1.1. SwiftStack arkitektur visade sig vara en flaskhals även då SwiftStack jämfördes med GlusterFS. Detta yttrade sig i en hög CPU utnyttjandegrad, vilket framgår av mätningarna i avsnittet 3.3.3.3. Ytterligare ett arbete kring dataarkivering undersöktes (se avsnitt 3.3.1.2). I detta arbete utfördes mätningar kring skalbarheten för GlusterFS. Testet visade att GlusterFS uppnådde bättre skalbarhet vid mätningarna av enstaka klientservrar jämfört med när de kördes samtidigt. Dock behandlade detta arbete endast GlusterFS. Eftersom att det inte hittades direkta jämförelser gällande skalbarhet mellan GlusterFS och SwiftStack samt GlusterFS och Ceph, blev det svårt att bestämma vilken dataarkiveringslösning bland samtliga öppna distribuerade filsystem var lämpligast att välja.

Ännu en parameter som togs i beaktande när det gäller *Cold Storage* är dataskydd, speciellt när det kommer till säkerhetskopiering. Ceph löste detta med hjälp av replikering vilket klarar av datamängder av petabytestorleksordning [87] medan GlusterFS använder sig av RAID 1 metoden för att replikera data (se avsnitt 3.3.2.1). Det har visat sig att RAID har brister när det gäller förmåga att klara av att hantera datamängder av petabytestorleksordning. Detta mest på grund av att det kan ta upp till flera timmar och ibland upp till flera dagar, beroende på katastrofgraden, att återhämta sitt data ifall en lagringsenhet går ned. Vid sådana situationer påverkas även systemprestanda. Vidare kan konstateras att ju mer data som lagras, desto mer växer lagringssystemet och desto fler lagringsenheter krävs det. Detta i sin tur innebär ökad risk för att fler diskar går ned samtidigt. RAID 1 klarar av att återhämta data om upp till två diskar går ned samtidigt. Dessutom behöver RAID allokera mycket ledigt utrymme för återställningsåtgärder, vilket i sin tur innebär att detta lagringsutrymme är oanvändbart. Ur ett ekonomiskt perspektiv löser RAID säkerhetskopiering på ett ineffektivt sätt gentemot replikeringsmetoden som Ceph använder sig av [77, 88]. Vidare bygger RAID system på en array av diskar som kräver att alla diskar har exakt likadana egenskaper, vilket inte är fallet när det gäller Ceph [77, 88]. Ytterligare ett arbete kring dataskydd gällande säkerhetskopiering undersöktes (se avsnitt 3.3.2.2). I detta arbete utfördes mätningar kring effektiviteten vid säkerhetskopiering för SwiftStack. Testet visade brister vid säkerhetskopiering av data i form av längre dataöverföringstid samt kopiering av redundant data. Dock behandlade detta arbete endast SwiftStack. Eftersom att det inte hittades direkta jämförelser gällande säkerhetskopiering mellan SwiftStack och GlusterFS samt SwiftStack och Ceph, blev det svårt att bestämma vilken säkerhetskopieringslösning bland samtliga öppna distribuerade filsystem var lämpligast att välja.

Vidare gällande återställning inom dataskydd, undersöktes återställningstiden för SwiftStack och GlusterFS (se avsnitt 3.3.2.3). Denna jämförelse visade även en uppenbart bättre återställningstid för SwiftStack, vilket är en bra egenskap för att återuppta systemet fortare efter en katastrofåterställning (disaster recovery). Ytterligare ett arbete kring dataskydd gällande återställning undersöktes (se avsnitt 3.3.2.4). I denna doktorsavhandlning utfördes mätningar gällande återställningstiden för Ceph. Enligt författaren visade testerna ovanligt lång återställningstid. Detta motiveras genom att författaren nämner att då mätningarna utfördes, användes ett relativt litet system med fåtal lagringsenheter medan i verkligheten liknande system används av tusentals lagringsenheter som sprider ut sina kopior till många fler lagringsenheter, och därmed blir replikeringstiden kortare.

Dock behandlade detta arbete endast Ceph. Eftersom att det inte hittades direkta jämförelser gällande återställningstiden mellan Ceph och GlusterFS samt Ceph och SwiftStack, blev det svårt att bestämma vilken återställningslösning bland samtliga öppna distribuerade filsystem var lämpligast att välja.

Då det fattades direkta jämförelser för dataarkivering (skalbarhet) och dataskydd (säkerhetskopiering och återställning) mellan samtliga öppna distribuerade filsystem, blev det svårt att avgöra vilken lösning var bäst att tillämpa för *Cold Storage*.

Ytterligare egenskaper kring samtliga användningsfall framgår av tabell 21, där Ceph hade de flesta egenskaperna jämfört med SwiftStack och GlusterFS.

Å andra sidan jämförelse mellan kommersiella lösningar visade inga större skillnader i varken prestanda eller pris då det gäller objektbaserad lagring. Dock uppstod det skillnader i pris när normallagring och snabblagring togs i beaktande, vilket framgår av tabell 25. Detta berodde på att olika företag som tillhandahåller datalagringslösningar, erbjuder olika egenskaper för olika lösningar. Därmed kan det slutgiltiga priset per GB/månad variera. För *Virtual Machine Storage*, är snapshots en viktig egenskap. Exempelvis erbjuder Microsoft snapshots som en tillkommande kostnad per GB/månad. Denna egenskap kan påverka det slutgiltiga priset beroende på antal snapshots som tas samt lagringsvolymen som ett snapshot ska täcka [86]. När det gäller *High Performance Storage* erbjuder samtliga företag lösningar med garanterat antal IOPS. Detta mest för att uppfylla kraven som ställs på olika högfrekventa databaser där tusentals läsningar och skrivningar sker i sekunden. Ytterligare kostnader kan tillkomma beroende på antalet IOPS som önskas.

Ur ett ekonomiskt perspektiv blev öppna DFS en lönsammare lösning att tillämpa, vilket framgår av tabell 25. Den totala kostnaden skulle kunna sänkas ytterligare om billigare varianter av HDD köps när det gäller icke-högprestandabaserad lagring. Ytterligare faktorer som kan påverka den slutgiltiga kostnaden är att samtliga öppna DFS ställer inga särskilda krav för vilken hårdvara de ska köras på samt att övriga egenskaper bland annat snapshotting är redan inbyggda i mjukvaran. En ekonomisk nackdel med öppna DFS är att tjänsteleverantören måste ha utrymme för placering av utrustning, exempelvis en serverhall eller att hyra en rack för att placera sin hårdvara. Detta kan skapa ytterligare en kostnad. Ännu en nackdel kan vara möjlighet att replikera data geografiskt det vill säga på ett avlägset ställe, exempelvis på en annan kontinent. På så sätt skulle tjänsteleverantören kunna erbjuda en tjänst med högre säkerhetsgrad. Dock skulle denna tjänst skapa ytterligare kostnader.

Att implementera egen lösning kan innebära vissa miljökonsekvenser i form av ökad elförbrukning då denna lösning delvis bygger på hårdvara som i sin tur är energikrävande. Vidare måste egen lösning administreras vilket förutsätter anställning av en administratör. Om en kommersiell lösning skulle implementeras istället, skulle företaget inte haft behov av varken hårdvara som kräver resurser i form av el samt regelbundet utbyte eller någon intern administratör. I så fall skjuts alla dessa oundvikliga administrationsuppgifter vidare till en annan geografisk ort där tjänsten erbjuds ifrån. Detta för att miljöfrågor inte endast är lokala utan även globala.

Vid implementation av egen lösning, kan företaget se till att uppdatera mjukvaran regelbundet vilket oftast är mer optimerad än föregående version och därmed kräver mindre resurser (exempelvis lägre CPU utnyttjandegrad). Faktum att skapa egna lösningar, skulle innebära fler arbetsplatser vilket skulle gynna både staten och de anställda.

Det är värt att nämna att uppdaterade versioner av dessa undersökta DFS kommer ut regelbundet vilket innebär förbättringar av prestanda, skalbarhet, tillförlitlighet samt nya funktioner. Med tanke på att vissa av arbeten som analyserades var upp till 5 år gamla, bör dessa jämförelser tas med en nypa salt.

5 Slutsats

Utifrån de kraven om prestanda, kostnader och dataskydd undersöktes olika användningsfall för öppna distribuerade filsystem. Kommersiella distribuerade filsystem undersöktes endast för att ge en överblick över hur de etablerade företagens datalagringslösningar fungerar.

I de testerna som jämfördes från tidigare arbeten kring *High Performance Storage*, visade sig att Ceph uppnådde bäst resultat i jämförelse med samtliga öppna distribuerade filsystem. Även för *Virtual Machine Storage*, påvisade undersökta mätningar att Ceph hade bäst resultat för samtliga egenskaper som kännetecknar *Virtual Machine Storage.* För det tredje undersökta användningsfallet *Cold Storage* saknades dock direkta jämförelser mellan GlusterFS, SwiftStack och Ceph. Detta blev ett hinder i strävan efter att bestämma den kompletta lösningen som arbetsgivaren var ute efter. Trots att det är svårt att komma fram till en komplett lösning, kan Ceph implementeras för *High Performance Storage* och *Virtual Machine Storage*. De flesta egenskaper som Ceph kännetecknades av visade att det går att implementera denna öppna distribuerade filsystemet på vilken hårdvara samt med vilka hårddiskar som helst. Detta innebär att tjänsteleverantören kan köpa de billigaste hårdvaran samt hårddiskar för att kunna dra ner på kostnaderna.

Kostnadsuppskattningen som gjordes för kommersiella- och öppna distribuerade filsystem, kan användas som ett underlag för att påpeka vilka typer av kostnader som tillkommer ifall ett distribuerat filsystem ska implementeras. Examensarbetarna hoppas på att detta arbete kan bidra till ökad förståelse kring samtliga distribuerade filsystem som undersökts.

Någon vidareutveckling av denna undersökning hade förbättrats genom att få tillgång till utrustning för att utföra praktiska mätningar på de senaste versionerna av de öppna distribuerade filsystemen. Mätningar för samtliga viktiga parametrar (skalbarhet, prestandaisolering, läs- och skrivtester, säkerhetskopiering och återställning) kan då utföras och direkta jämförelser och analyser av samtliga distribuerade filsystem skulle ge en jämnare och bättre överblick av vilket distribuerat filsystem som skulle vara lämpligt att användas som en komplett lösning.

Denna undersökning som genomfördes i examensarbetet kan användas som en grund för tjänsteleverantörer för att reda ut vilket öppet distribuerat filsystem som är lämpligast utifrån tjänsteleverantörernas behov.

Källförteckning

[1] – Big Data,
URL: https://www.ibm.com/big-data/us/en/
Hämtad: 2016-06-23

[2] – Peter M. Chen, David A. Patterson ”Storage Performance – Metrics and Benchmarks”, University of California, Berkeley, CA, USA,
URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.47.2136&rep=rep1&type=pdf
Hämtad: 2016-06-18

[3] – Lucas Scott Hindman, ”DESIGNING RELIABLE HIGH-PERFORMANCE STORAGE SYSTEMS FOR HPC ENVIRONMENTS”, Boise State University, Boise, Idaho, USA,
URL: http://scholarworks.boisestate.edu/cgi/viewcontent.cgi?article=1216&context=td
Publicerad: Augusti 2011, Hämtad: 2016-04-02

[4] – W. Curtis Preston, ”Data Protection Strategies In Today’s Data Center”, Oracle, 2012, Truth-InIt, Oceanside, CA, USA,
Publicerad: Januari 2012, Hämtad: 2016-06-29

[5] – Tandberg Data, ”Guide to Data Protection Best Practices”, Tandberg, 2011,
Publicerad: 2011, Hämtad: 2016-07-01

[6] – Ayman Zalet, “Vklass datalagring”, KTH STH, Stockholm, SWE,
Publicerad: Januari 2016, Hämtad: 2016-04-02

[7] – Ceph.com, “Ceph Block Device”, Ceph,
URL: http://docs.ceph.com/docs/jewel/rbd/rbd/
Hämtad: 2016-06-19.

[8] – Stonefly.com, ”What is File Level Storage vs. Block Level Storage?”, StoneFly,
URL: http://www.iscsi.com/resources/File-Level-Storage-vs-Block-Level-Storage.asp
Hämtad: 2016-07-01

[9] – Jon Tate, Pall Beck, Libor Miklas, Hector Hugo Ibarra, Shanmuganathan Kumaravel, ”Introduction to Storage Area Networks”, IBM, Redbooks, nr. 7, 2016,
Publicerad: Januari 2016, Hämtad: 2016-06-27.

[10] – S.K Chang, ”Physical Structures”, University of Pittsburgh,Pittsburgh, PA, USA,
URL: http://people.cs.pitt.edu/~chang/156/08struct.html
Hämtad: 2016-06-28.

[11] - R.C. Daley, ”A General-Purpose File System For Secondary Storage”, Massachusetts Institute of Technology, Cambridge, MA, USA,
URL: http://www.multicians.org/fjcc4.html
Hämtad: 2016-06-29.

[12] - Canonical, ”How en object store defers fom file and block storage”, Ubuntu,
URL: https://insights.ubuntu.com/2015/05/18/what-are-the-different-types-of-storage-block-object-and-file/
Publicerad: 2015-05-18, Hämtad: 2016-06-30.

[13] – Seagate, “What is NAS and why is NAS important for small business?”, Seagate Technology,
URL: http://www.seagate.com/gb/en/tech-insights/what-is-nas-master-ti/
Publicerad: 2016, Hämtad: 2016-06-29

[14] - Dell Product Group, ”Object Storage: A Fresh Approach to Long-Term File Storage”, Dell, 2010,
Publicerad: Maj 2010, Hämtad: 2016-06-29

[15] – Dr. Brent Welch, “Object Storage Technology”, SNIA Education, 2013,
Publicerad: 2013, Hämtad: 2016-06-28

[16] - Duy Le, Hai Huang, Haining Wang, ” Understanding Performance Implications of Nested File Systems in a Virtualized Environment”, The College of William and Mary, Williamsburg, VA, USA,
URL: https://www.eecis.udel.edu/~hnw/paper/fast12.pdf
Hämtad: 2016-07-01.

[17] - IBM Cloud, ”What is Object Storage?”, IBM,
URL: https://www.ibm.com/cloud-computing/object-storage
Hämtad: 2016-07-01.

[18] – Dell DX Object Storage Platform, ”What is object storage?”, Dell,
URL: https://www.ibm.com/cloud-computing/object-storage
Hämtad: 2016-07-02.

[19] – Sage A. Weil, ”Ceph: Reliable, Scalable and High-Performance Distributed Storage”, Ceph,
URL: http://ceph.com/papers/weil-thesis.pdf
Publicerad: December 2007, Hämtad: 2016-05-10

[20] – Sage A. Weil, Scott A. Brandt, Ethan L. Miller, Carlos Maltzahn, “CRUSH: Controlled, Scalable, Decentralized Placement of Replicated Data”, Ceph,
URL: http://ceph.com/papers/weil-crush-sc06.pdf
Publicerad: November 2006, Hämtad: 2016-05-10

[21] – Gluster Docs, “GlusterFS Documentation”, GlusterFS,
URL: http://gluster.readthedocs.io/en/latest/
Hämtad: 2016-05-10

[22] – SwiftStack.com, “Powering the world’s largest storage clouds”, SwiftStack,
URL: https://www.swiftstack.com/openstack-swift/
Hämtad: 2016-05-10

[23] – Google Cloud Platform, “Powerful & Simple Storage”, Google,
URL: https://cloud.google.com/storage/
Hämtad: 2016-06-20

[24] – Google Cloud Platform, “Storage Classes”, Google,
URL: https://cloud.google.com/storage/docs/storage-classes
Hämtad: 2016-06-20

[25] – Amazon AWS, “Amazon S3”, Amazon
URL: https://aws.amazon.com/s3/
Hämtad: 2016-06-20

[26] – Amazon AWS, “Amazon S3 Storage Classes”, Amazon,
URL: https://aws.amazon.com/s3/storage-classes/
Hämtad: 2016-06-20

[27] – Microsoft Azure Storage, “Enorm skalbar molnlagring för dina program”, Microsoft,
URL: https://azure.microsoft.com/sv-se/services/storage/
Hämtad: 2016-06-21

[28] – Microsoft Azure Storage, “Blob Storage”, Microsoft,
URL: https://azure.microsoft.com/sv-se/services/storage/blobs/
Hämtad: 2016-06-21

[29] - Microsoft Azure Storage, “Products pricing”, Microsoft,
URL: https://azure.microsoft.com/sv-se/pricing/details/storage/
Hämtad: 2016-06-21

[30] – Microsoft Azure Storage, ”Replication”, Microsoft,
URL: https://azure.microsoft.com/sv-se/documentation/articles/storage-redundancy/
Publicerad: 2016-06-23, Hämtad: 2016-06-28

[31] – Dell EMC Glossary, ”Cold Data Storage”, Dell EMC,
URL: https://www.emc.com/corporate/glossary/cold-data-storage.htm
Hämtad: 2016-07-04

[32] – Amazon AWS, “Cold Storage”, Amazon,
URL: https://aws.amazon.com/glacier/
Hämtad: 2016-06-20

[33] – IBM Storage, ”Data archiving and improving application efficiency”, IBM,
URL: https://www-01.ibm.com/software/data/optim/manage-data-growth/index.html
Hämtad: 2016-07-05

[34] – Mike Cholewa, Niraj Desai, Drew Peterson, “Data Archiving: Foundation Capabilities for Compliance and Cost Optimization”, IBM, 2009,
Publicerad: April 2009, Hämtad: 2016-07-01

[35] – Glen Robinson, Attila Narin, Chris Elleman, ”Using Amazon Web Services for Disaster Recovery”, Amazon, 2014,
Publicerad: Oktober 2014, Hämtad: 2016-07-05

[36] - Cisco, “Disaster Recovery: Best Practices”, Cisco Systems, 2008,
Publicerad: 2008, Hämtad: 2016-07-01

[37] – Microsoft Technet, ”Switchovers and Failovers”, Microsoft,
URL: https://technet.microsoft.com/en-us/library/dd298067(v=exchg.150).aspx
Publicerad: 2015-12-02, Hämtad: 2016-07-06

[38] – MarkLogic Server, “Scalability, Availability, and Failover Guide”, MarkLogic Corporation, 2015,
Publicerad: Februari 2015, Hämtad: 2016-07-03

[39] – Ceph.com, “Placement Group”, Ceph,
URL: http://docs.ceph.com/docs/master/rados/operations/placement-groups/
Hämtad: 2016-06-23

[40] - Benjamin Depardon, Gaël Le Mahec, Cyril Séguin, ”Analysis of Six Distributed File Systems”, Laboratoire MIS, Université de Picardie Jules Verne, Amiens, FRA,
URL: https://hal.inria.fr/hal-00789086/file/a_survey_of_dfs.pdf
Publicerad: 2013-02-15, Hämtad: 2016-05-14

[41] – Alapan Mukherjee, ”Benchmarking Hadoop performance on different distributed storage systems”, Aalto University, Esbo, FIN,
URL: https://aaltodoc.aalto.fi/bitstream/handle/123456789/17713/master_Mukherjee_Alapan_2015.pdf
Publicerad: 2015-06-19, Hämtad: 2016-06-19

[42] - Da Zheng, Randal Burns, A. S. Szalay, ”Toward Millions of File System IOPS on Low-Cost, Commodity hardware”, Johns Hopkins University, Baltimore, MD, USA,
URL: http://www.cs.jhu.edu/~zhengda/sc13.pdf
Publicerad: November 2013, Hämtad: 2016-07-09.

[43] - Sapan Bhatia, Murtaza Motiwala, Wolfgang Mühlbauer, Vytautas Valancius, Andy Bavier, Nick Feamster, Larry Peterson, Jennifer Rexford, ”Hosting Virtual Networks on Commodity Hardware”, Princeton University, Princeton, NJ, USA,
URL: https://www.cs.princeton.edu/~jrex/papers/trellis07.pdf
Publicerad: 2009, Hämtad: 2016-07-05

[44] – Amazon AWS, ”An Introduction to High Performance Computing on AWS”, Amazon, 2015,
Publicerad: Augusti 2015, Hämtad: 2016-05-22

[45] – Stephen J. Bigelow, ”An Introduction to High Performance Computing on AWS”, SUN Microsystems, 2009,
Publicerad: 2009, Hämtad: 2016-05-20

[46] – Jacob Gorm Hansen, Eric Jul, “Scalable Virtual Machine Storage using Local Disks”, VMWare, Bell Laboratories, 2008,
Publicerad: 2008, Hämtad: 2016-07-10

[47] - Rouven Krebs, Christof Momm, Samuel Kounev, “Architectural Concerns in Multi-Tenant SaaS Applications”, SAP AG, Karlsruhe Institute of Technology, GER, 2013,
Publicerad: 2013, Hämtad: 2016-05-17

[48] – Oracle Documentation, ”What is a Snapshot?”, Oracle,
URL: http://docs.oracle.com/cd/A84870_01/doc/server.816/a76959/mview.htm
Hämtad: 2016-07-10.

[49] – IBM developerWorks, “Understanding and exploiting snapshot technology for data protection”, IBM,
URL: https://www.ibm.com/developerworks/tivoli/library/t-snaptsm1/
Publicerad: 2006-04-26, Hämtad: 2016-05-03

[50] – IBM Storage, ”IBM Data Deduplication Strategy and Operations ”, IBM,
Hämtad: 2016-07-12

[51] – Thomas Rivera, “UNDERSTANDING DATA DEDUPLICATION”, SNIA Education, 2009,
Publicerad: 2009, Hämtad: 2016-07-05

[52] – Dell EMC Glossary, ”Data deduplication”, Dell EMC,
URL: http://www.emc.com/corporate/glossary/data-deduplication.htm
Hämtad: 2016-07-17

[53] – Acronis, “How Deduplication Benefits Companies of All Sizes”, Acronis, 2009,
Publicerad: 2009, Hämtad: 2016-07-12

[54] – Arun Raman, “NetApp Thin Provisioning”, NetApp, 2007,
Publicerad: Maj 2007, Hämtad: 2016-07-13

[55] – Peter Williams, ”Storage Thin Provisioning Explained”, HP, 2010, Bloor Research, London, UK,
Publicerad: 2010, Hämtad: 2016-04-23

[56] - Martin Lnenicka, Jitka Komarkova, Eva Milkova, “Performance Testing of Cloud Storage while Using Spatial Data”, University of Pardubice, University of Hradec Kralove, *Konferens,* 2012, Barcelona, SPA,
Publicerad: 2012, Hämtad: 2016-06-29

[57] – Google Cloud Platform, “FAQ”, Google,
URL: https://cloud.google.com/storage/docs/faq
Publicerad: 2016-06-27, Hämtad: 2016-07-01

[58] – Amazon AWS, “Introduction to Amazon S3”, Amazon,
URL: http://docs.aws.amazon.com/AmazonS3/latest/dev/Introduction.html
Hämtad: 2016-07-08

[59] – M D Poat, J Lauret, W Betts, “POSIX and Object Distributed Storage Systems Performance Comparison Studies With RealLife Scenarios in an Experimental Data Taking Context Leveraging OpenStack Swift & Ceph”, *Konferens,* 2015, CHEP2015, Okinawa, JAP,
Publicerad: 2015, Hämtad: 2016-05-16

[60] – Scientific Linux, “Scientific Linux”, CERN, Fermilab,
URL: https://www.scientificlinux.org/
Hämtad: 2016-05-20

[61] – OpenStack Documentation, “Large Object Support”, OpenStack,
URL: http://docs.openstack.org/developer/swift/overview_large_objects.html
Hämtad: 2016-05-18

[62] – Ceph Documentation, “RBD – Manage RADOS Block Device (RBD) Images”, Ceph,
URL: http://docs.ceph.com/docs/master/man/8/rbd/
Hämtad: 2016-05-18

[63] – Grid5000.fr, “Grid5000: Home”, Grid’5000,
URL: https://www.grid5000.fr/mediawiki/index.php/Grid5000:Home
Hämtad: 2016-05-20

[64] - Sogand Shirinbab, Lars Lundberg, David Erman, ”Performance Evaluation of Distributed Storage Systems”, Blekinge Tekniska Högskola, Karlskrona, SWE,
URL: http://www.gluster.org/pipermail/gluster-users/attachments/20130122/21bf192d/attachment.obj
Publicerad: December 2013, Hämtad: 2016-06-15

[65] – Ondrej Famera, “Cloud backend deployment at Faculty of informatics MU”, Faculty of Informatics Masaryk University, Brno, CZE,
URL: http://is.muni.cz/th/324964/fi_m/thesis.pdf
Publicerad: 2013, Hämtad: 2016-05-17

[66] – Redhat Documentation, “The Ext4 File System”, Redhat,
URL: https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/6/html/Storage_Administration_Guide/ch-ext4.html
Hämtad: 2016-05-20

[67] – Oracle Technology Network, “Btrfs – The Next Generation File System for Linux”, Oracle,
URL: http://www.oracle.com/technetwork/server-storage/linux/technologies/btrfs-overview-1898045.html
Hämtad: 2016-05-20

[68] – A. Tanenbaum, H. Bos, ”Modern Operating Systems”, nr. 4, 2015, s. 300-302,
Publicerad: 2015, Hämtad: 2016-07-20

[69] – A. Tanenbaum, H. Bos, ”Modern Operating Systems”, nr. 2, 2001, s. 430-447,
Publicerad: 2001, Hämtad: 2016-06-28

[70] - Keiichi Shima, Nam Dang, “Indexes for Distributed File/Storage Systems as a Large Scale Virtual Machine Disk Image Storage in a Wide Area Network”, IIJ Innovation Institute, Tokyo Institute of Technology, Tokyo, JAP,
URL: http://member.wide.ad.jp/~shima/publications/20120924-dfs-performance.pdf
Publicerad: 2012-09-24, Hämtad: 2016-05-18

[71] – Ceph Documentation, “Ceph Block Device”, Ceph,
URL: http://docs.ceph.com/docs/master/rbd/rbd/
Hämtad: 2016-05-18

[72] – Gluster Docs, “GlusterFS General FAQ”, GlusterFS,
URL: http://www.gluster.org/community/documentation/index.php/GlusterFS_General_FAQ
Hämtad: 2016-05-18

[73] – Ceph Documentation, “Hardware Recommendations”, Ceph,
URL: http://docs.ceph.com/docs/jewel/start/hardware-recommendations/
Hämtad: 2016-05-18

[74] – SwiftStack.com, “Enterprise-Ready Object Storage Software”, SwiftStack,
URL: https://www.swiftstack.com/product
Hämtad: 2016-06-14

[75] – OpenStack Documentation, ”Replication”, OpenStack,
URL: http://docs.openstack.org/developer/swift/overview_replication.html
Publicerad: 2016-07-20, Hämtad: 2016-07-01

[76] – Divya Muntimadugu, ”Gluster File System Administration Guide 3.3.0 – Using Gluster File System”, GlusterFS, 2012,
Publicerad: 2012-04-05, Hämtad: 2016-07-08

[77] – Karan Singh, ”Learning Ceph”, 2015, s. 15-16,
Publicerad: Januari 2015, Hämtad: 2016-05-18

[78] – OpenStack Documentation, ”Administration Guides – Defining Storage Policies”, OpenStack,
URL: http://docs.openstack.org/developer/swift/admin_guide.html
Publicerad: 2016-07-20, Hämtad: 2016-07-05

[79] – Gluster Docs, ”GlusterFS Volume Snapshot”, GlusterFS,
URL: http://www.gluster.org/community/documentation/index.php/Features/Gluster_Volume_Snapshot
Publicerad: 2014-09-25, Hämtad: 2016-06-14

[80] – Ceph Documentation, “Snapshots”, Ceph,
URL: http://docs.ceph.com/docs/hammer/rbd/rbd-snapshot/
Hämtad: 2016-05-17

[81] - Alexandre Beslic, Reda Bendraou, Julien Sopena, Jean-Yves Rigolet, "Towards a solution avoiding Vendor Lock-in to enable Migration Between Cloud Platforms", Pierre and Marie Curie University, *Konferens,* 2013, MDHPCL, FL, USA,
Publicerad: 2013, Hämtad: 2016-07-08

[82] – Lonestatistik.se, “Systemadministratör”, Lönestatistik,
URL: http://www.lonestatistik.se/loner.asp/yrke/Systemadministrator-7811
Hämtad: 2016-07-07

[83] – EON.se, “Teckna elavtal”, EON,
URL: https://www.eon.se/foeretag/handla/teckna-elavtal/anpassa-elavtal.html
Hämtad: 2016-07-14

[84] – Google Cloud Storage, “Storage Options”, Google,
URL: https://cloud.google.com/compute/docs/disks/
Publicerad: 2016-06-27, Hämtad: 2016-07-20

[85] – Amazon AWS, “Amazon EBS pricing”, Amazon,
URL: https://aws.amazon.com/ebs/pricing/
Hämtad: 2016-07-20

[86] – Microsoft Azure Storage, “Azure Storage Pricing”, Microsoft,
URL: https://azure.microsoft.com/en-us/pricing/details/storage/
Hämtad: 2016-07-20

[87] – IBM developerWorks, “Ceph: A Linux petabyte-scale distributed file system”, IBM,
URL: http://www.ibm.com/developerworks/library/l-ceph/
Publicerad: 2010-06-07, Hämtad: 2016-05-20

[88] – Redhat Storage, ”RAID VERSUS REPLICATION – The end of RAID as you know it”, Redhat, 2015,
Publicerad: 2015, Hämtad: 2016-05-21

[89] – Gluster, “Cloud Storage for the Modern Data Center”,GlusterFS, 2011,
Publicerad: 2011, Hämtad: 2016-07-13

[90] – Liontech.com, “HP 5900AF-48XG-4QSFP+”, Liontech,
URL: http://www.liontech.se/partdetail.aspx?q=p:4525113
Hämtad: 2016-07-12

[91] – Multritronic.se, “HP A58x0AF 650W”, Multitronic,
URL: https://www.multitronic.se/sv/products/1127740/hp-a58x0af-650w-dc-power-supply
Hämtad: 2016-07-12

[92] – QSFPS.com, “HP X240 10G SFP+ to SFP+ 3m”, QSFPS,
URL: http://www.qsfps.com/jd097c
Hämtad: 2016-07-12

[93] – Warehouse1.com.au, “HP R7KVA UPS 4U IEC-32A HV Intl Kit”, Warehouse1,
URL: http://www.warehouse1.com.au/epages/shop.sf/?ObjectPath=/Shops/warehouse1/Products/000000000001736184
Hämtad: 2016-07-12

[94] – Misco.se, “HP DL380 Gen9 12LFF CTO Server”, Misco,
URL: http://www.misco.se/product/product.aspx?P_ItemId=11128067&StartIndex=11&SL_SectionId=3205935
Hämtad: 2016-07-12

[95] – CPLonline.com.au, “HP DL380 Gen9 E5-2640v3 Kit”, CPLonline,
URL: http://cplonline.com.au/hp-dl380-gen9-e5-2640v3-kit.html
Hämtad: 2016-07-12

[96] – LambdaTek.com, ”HP DL380 Gen9 E5-2640v3 FIO Kit”, LambdaTek,
URL: http://www.lambda-tek.com/HP-719049-L21~sh/B1940628
Hämtad: 2016-07-12

[97] – Ebuyer.com, ”HP 16GB 2Rx4 PC4-2133P-R Kit”, Ebuyer,
URL: http://www.ebuyer.com/664853-hpe-16gb-2rx4-pc4-2133p-r-kit-726719-b21
Hämtad: 2016-07-12

[98] – Dustinhome.se, ”Seagate Mobile HDD 2048GB 2.5" SATA-600 5400rpm”, Dustin,
URL: https://www.dustinhome.se/product/5010910573/mobile-hdd
Hämtad: 2016-07-12

[99] – Conrad.se, ”SSD-Hårddisk 2.5" Intel S3500 Box 800 GB”, Conrad,
URL: https://www.conrad.se/?websale8=conrad-swe&pi=1398339&ws_tp1=cp&ref=pricerunner&subref=1398339&utm_source=pricerunner&utm_medium=feed&utm_campaign=pricerunner_feed&utm_content=1398339
Hämtad: 2016-07-12

[100] – CPLonline.com.au, “HP FlexFabric 10Gb 2P 534FLR-SFP+”, CPLonline,
URL: http://cplonline.com.au/hp-flexfabric-10gb-2p-534flr-sfp-adptr-700751-b21.html
Hämtad: 2016-07-12

[101] – Shi.com, ”HP H240ar FIO Smart HBA”, Shi,
URL: https://www.shi.com/Products/ProductDetail.aspx?SHISystemID=ShiCommodity&ProductIdentity=29931849
Hämtad: 2016-07-12

[102] – Kontorsmagasinet.se, ”HP 12Gb DL380 Gen9 SAS Expander Card”, Kontorsmagasinet,
URL: http://www.kontorsmagasinet.se/hp-sas-expander-card-uppgraderingskort-till-p-52?from=gpla&gclid=CITrzpKv8M0CFegMcwodw68A4g
Hämtad: 2016-07-12

[103] – Manonit Kumar, “Characterizing the GlusterFS Distributed File System for Software Defined Networks Research”, New Brunswick Rutgers, The State University of New Jersey, NJ, USA,
URL: https://rucore.libraries.rutgers.edu/rutgers-lib/46377/PDF/1/
Publicerad: Januari 2015, Hämtad: 2016-09-05

[104] – Amina Mseddi, Mohammad Ali Salahuddin, Mohamed Faten Zhani, Halima Elbiaze, Roch H. Glitho, ”On Optimizing Replica Migration in Distributed Cloud Storage Systems”, Montreal, CA,
URL: https://arxiv.org/pdf/1509.01330v1.pdf
Publicerad: 2015-09-05, Hämtad: 2016-09-05

[105] – Statsskuld.se, ”Beräkning av social- och arbetsgivaravgift”, Statsskuld,
URL: https://statsskuld.se/jobb/berakna-nettolon
Hämtad: 2016-09-06

[106] – Ceph Documentation, “Architecture”, Ceph,
URL: http://docs.ceph.com/docs/master/architecture/
Hämtad: 2016-09-04

[107] – Kannan Ramchandran, “Distributed Data Storage Systems”, University of California, Berkeley, CA, USA,
URL: https://people.eecs.berkeley.edu/~kannanr/project_dss.html
Publicerad: 2015-07-21, Hämtad: 2016-09-13

[108] – StorPool Distributed Storage, “What is distributed storage system and why is it important?”, StorPool,
URL: https://storpool.com/blog/what-is-distributed-storage-system
Hämtad: 2016-09-13

[109] – Microsoft TechNet, “Distributed File System”, Microsoft,
URL: https://technet.microsoft.com/en-us/library/cc753479(v=ws.10).aspx
Publicerad: 2009-07-15, Hämtad: 2016-09-13

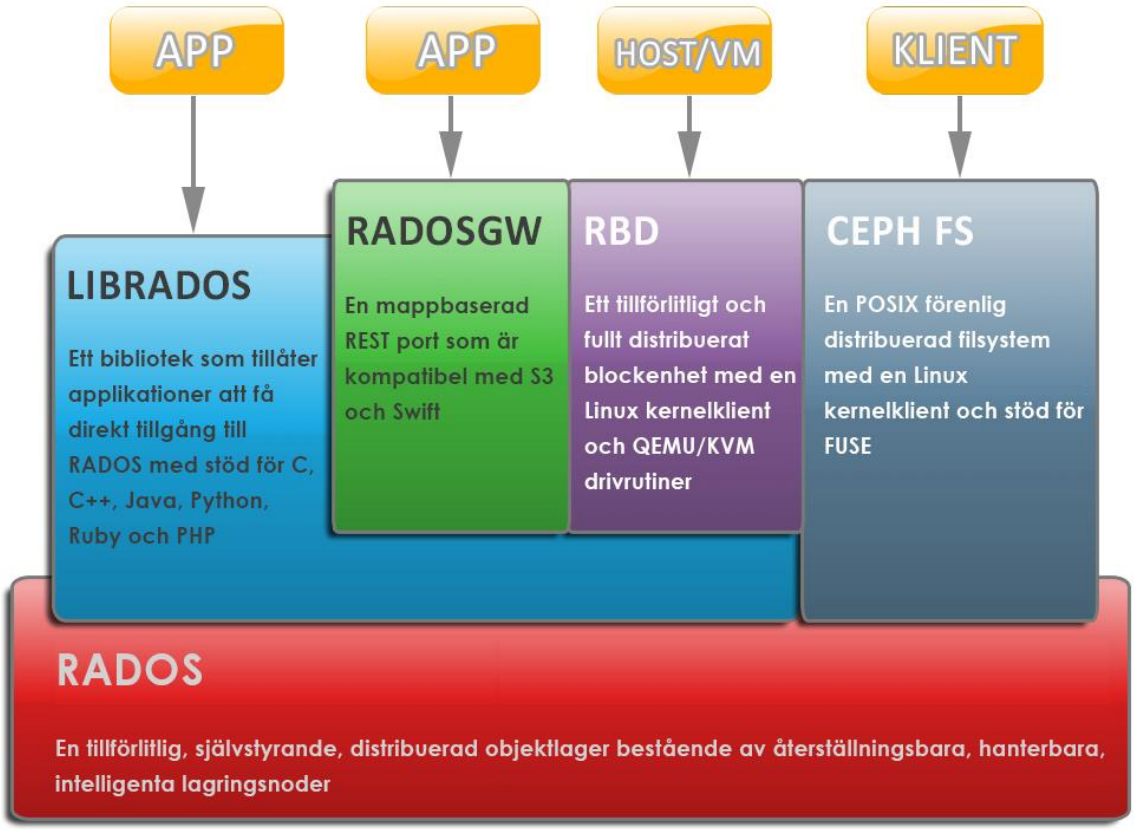
[110] - A. Tanenbaum, M. V. Steen, ”Distributed Systems, Principles and Paradigms”, nr. 2, 2007, s. 2-30, s. 491-540.
Publicerad: 2007, Hämtad: 2016-09-13

[111] – Greg Schulz, “Data Center I/O Performance Issues and Impacts”, StorageIO, 2006,
Publicerad: 2006-08-07, Hämtad: 2016-07-18

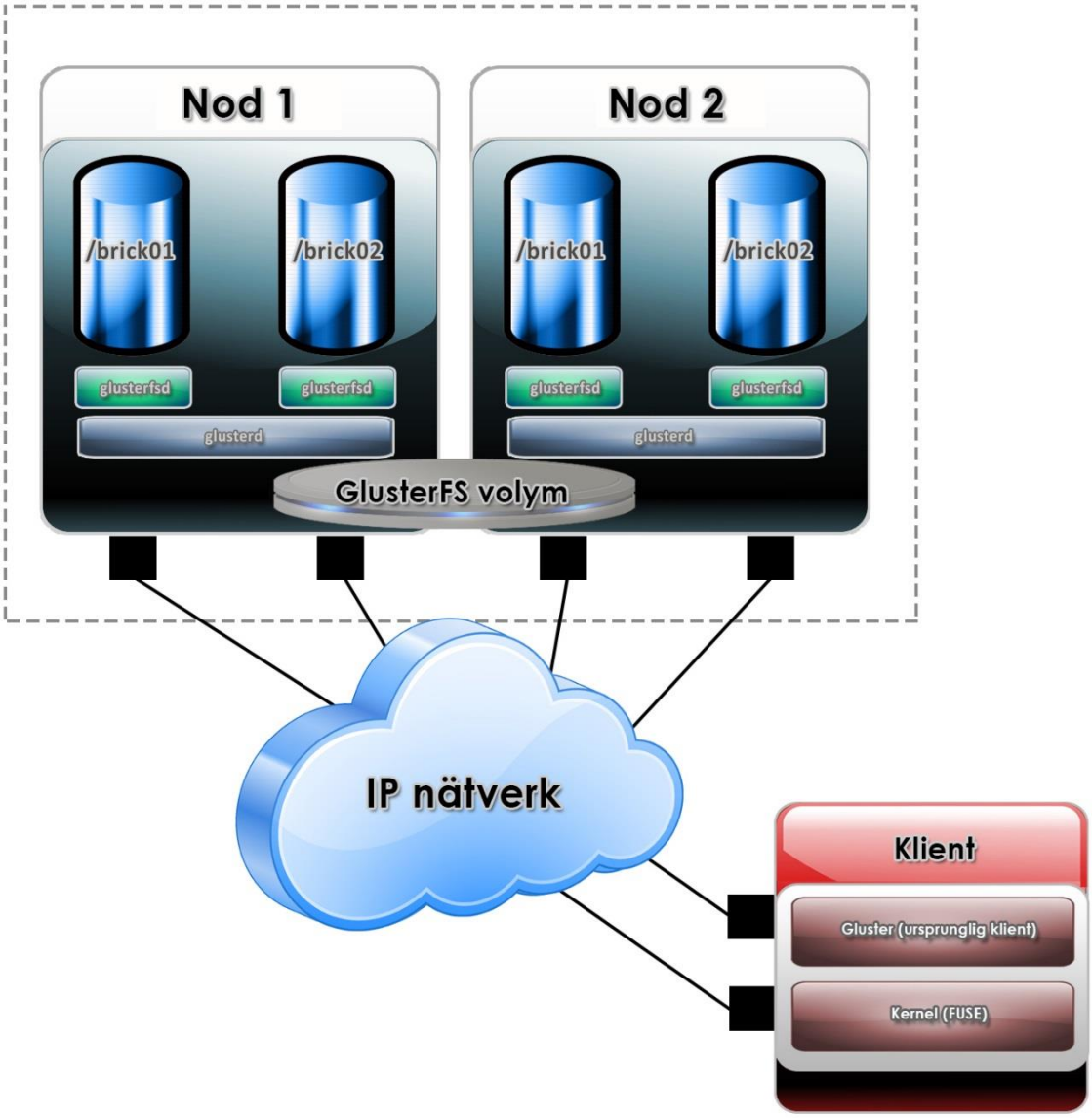
[112] – Dan Gibson, “Is Your Big Data Hot, Warm or Cold?”, IBM,
URL: http://www.ibmbigdatahub.com/blog/your-big-data-hot-warm-or-cold
Publicerad: 2012-06-01, Hämtad: 2016-07-19

Bilagor

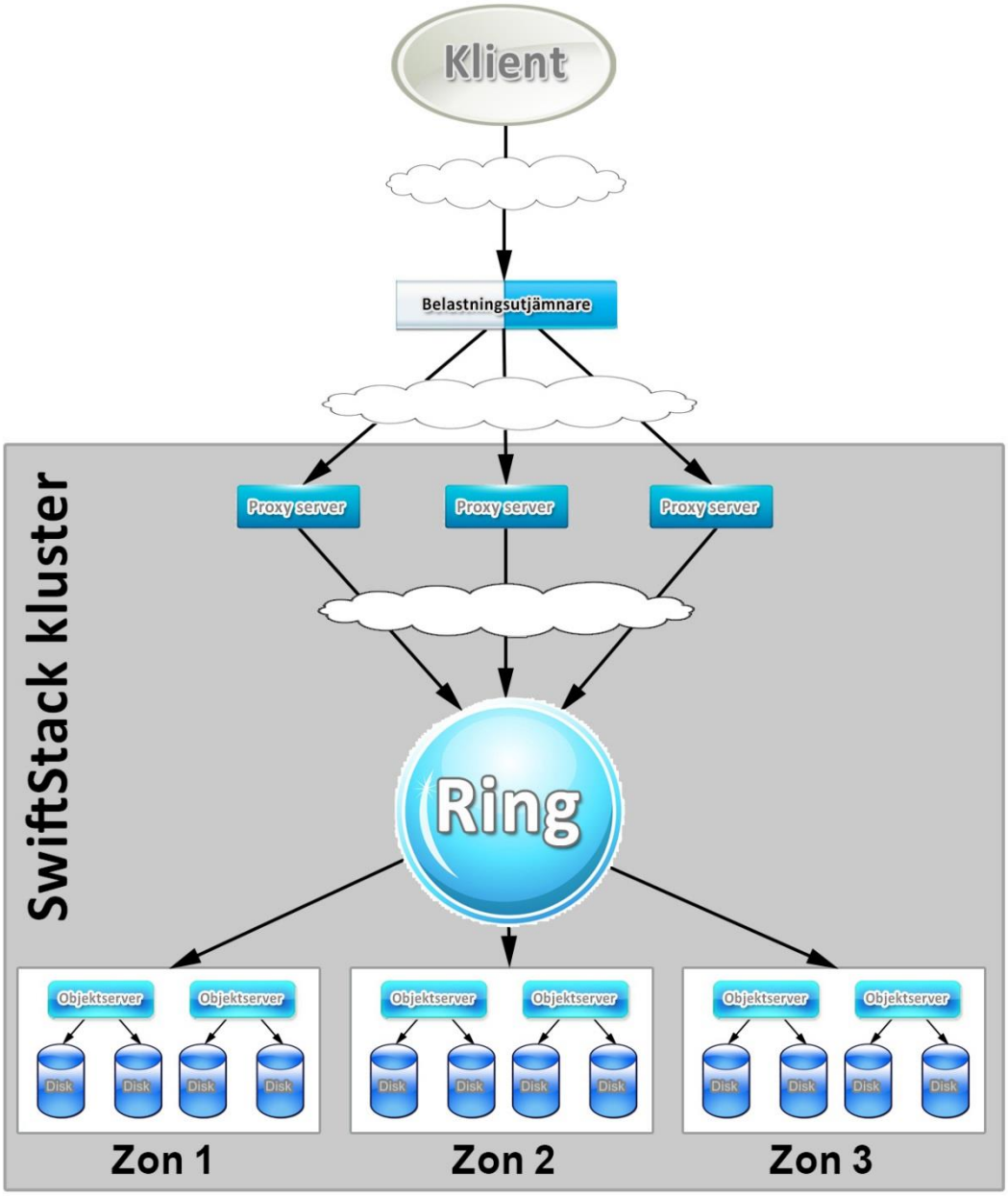
Bilaga 1: Arkitekturen för det distribuerade filsystemet Ceph [106].



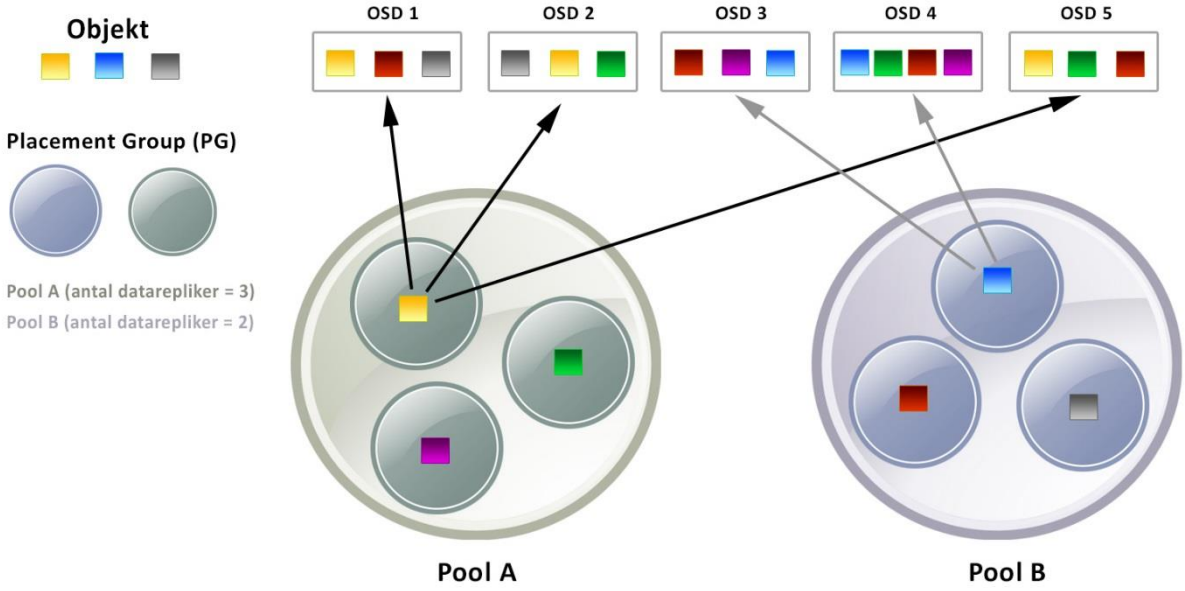
Bilaga 2: Arkitekturen för det distribuerade filsystemet GlusterFS.



Bilaga 3: Arkitekturen för det distribuerade filsystemet SwiftStack.



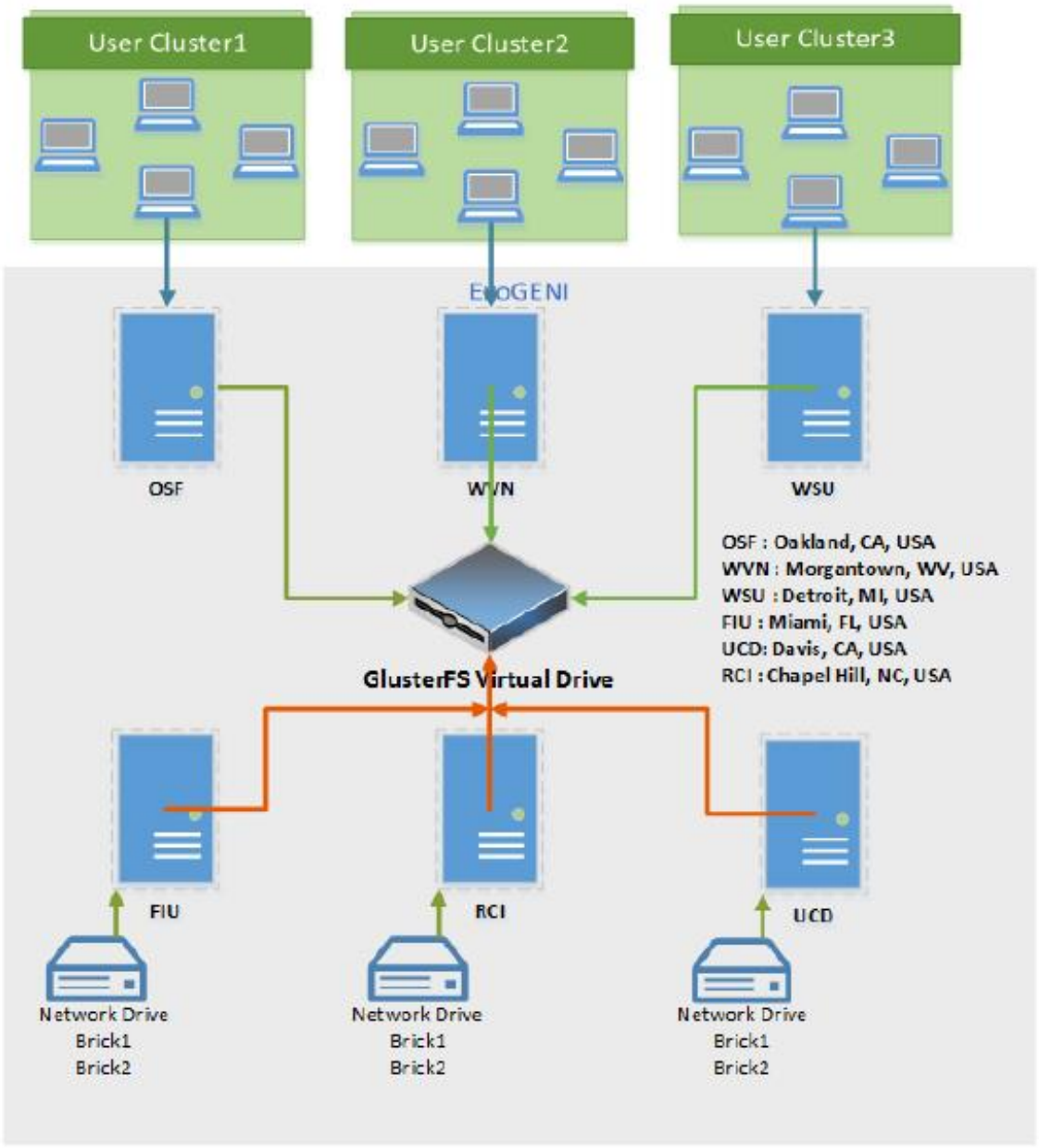
Bilaga 4: Illustration på hur data replikeras i det distribuerade filsystemet Ceph.



Bilaga 5: Figuren visar detaljerat resultat av den genomsnittliga uppladdnings- och nedladdningshastigheten på två platser. Filstorleken som användes vid läs- och skrivtesterna var 3,57 MB (se sidan 3) [56].

Testing CSP from location / time of test			Test 1 – 10:00	Test 2 – 11:00	Test 3 – 12:00	Test 4 – 13:00	Test 5 – 14:00	Test 6 – 15:00	Test 7 – 16:00
Upload speed	Location 1	Amazon	146,5 s; Ø 26,1 kB/s	145,2 s; Ø 26,3 kB/s	150,6 s; Ø 25,3 kB/s	149,3 s; Ø 25,5 kB/s	148,9 s; Ø 25,6 kB/s	149,1 s; Ø 25,5 kB/s	149,8 s; Ø 25,4 kB/s
		Google	140,5 s; Ø 26,6 kB/s	139,8 s; Ø 26,8 kB/s	140,6 s; Ø 26,7 kB/s	141,3 s; Ø 26,5 kB/s	142 s; Ø 26,3 kB/s	141,8 s; Ø 26,4 kB/s	141,9 s; Ø 26,4 kB/s
		Windows	137,9 s; Ø 27,3 kB/s	138,2 s; Ø 27,1 kB/s	138,1 s; Ø 27,1 kB/s	139 s; Ø 26,9 kB/s	138,1 s; Ø 27,1 kB/s	138,4 s; Ø 27 kB/s	138,5 s; Ø 27 kB/s
	Location 2	Amazon	82 s; Ø 46,6 kB/s	81,6 s; Ø 46,7 kB/s	81,7 s; Ø 46,7 kB/s	82 s; Ø 46,6 kB/s	82,7 s; Ø 46,1 kB/s	82,9 s; Ø 46 kB/s	83,2 s; Ø 45,8 kB/s
		Google	75,1 s; Ø 51,3 kB/s	74,9 s; Ø 51,5 kB/s	75,6 s; Ø 51 kB/s	76 s; Ø 50,7 kB/s	76,3 s; Ø 50,4 kB/s	76,9 s; Ø 50 kB/s	77,3 s; Ø 49,8 kB/s
		Windows	75,6 s; Ø 50,6 kB/s	76,5 s; Ø 49,9 kB/s	75,9 s; Ø 50,7 kB/s	76,1 s; Ø 50,5 kB/s	77,4 s; Ø 49,7 kB/s	77,6 s; Ø 49,6 kB/s	77,8 s; Ø 49,4 kB/s
Download speed	Location 1	Amazon	12,7 s; Ø 302,9 kB/s	12,9 s; Ø 301 kB/s	12,8 s; Ø 301,5 kB/s	12,8 s; Ø 301,4 kB/s	13 s; Ø 295,4 kB/s	13,3 s; Ø 289 kB/s	13,2 s; Ø 291,1 kB/s
		Google	12,8 s; Ø 301,1 kB/s	12,6 s; Ø 306,2 kB/s	12,4 s; Ø 310 kB/s	12,5 s; Ø 308,9 kB/s	12,7 s; Ø 302,8 kB/s	13 s; Ø 295,6 kB/s	13,1 s; Ø 293,5 kB/s
		Windows	12 s; Ø 317,7 kB/s	12,2 s; Ø 313,8 kB/s	12,1 s; Ø 316,9 kB/s	12 s; Ø 317,8 kB/s	12,1 s; Ø 318,6 kB/s	12,5 s; Ø 308,3 kB/s	12,9 s; Ø 300,9 kB/s
	Location 2	Amazon	8,1 s; Ø 471,6 kB/s	8 s; Ø 478,1 kB/s	8,3 s; Ø 460,6 kB/s	8,3 s; Ø 460,7 kB/s	8,5 s; Ø 451,8 kB/s	8,7 s; Ø 442,4 kB/s	8,9 s; Ø 430,5 kB/s
		Google	8,1 s; Ø 471,5 kB/s	8,4 s; Ø 455 kB/s	7,9 s; Ø 483,3 kB/s	8,1 s; Ø 471,3 kB/s	8,2 s; Ø 466,1 kB/s	8,3 s; Ø 460,5 kB/s	8,4 s; Ø 454,7 kB/s
		Windows	7,5 s; Ø 509,4 kB/s	7,8 s; Ø 489,8 kB/s	7,7 s; Ø 496,2 kB/s	7,8 s; Ø 489,4 kB/s	8,2 s; Ø 465,9 kB/s	8,1 s; Ø 471,3 kB/s	8,3 s; Ø 460 kB/s

Bilaga 6: Illustration på hur infrastrukturen såg ut där testerna utfördes i arbetet ”Characterizing the GlusterFS Distributed File System for Software Defined Networks Research” (se sidan 27) [103].



Bilaga 7: Tabellen visar den totala kostnaden för utrustningen som användes för att göra kostnadsuppskattningen. Valutakursen som användes för omvandling av priset för samtliga utrustningar till svenska kronor, togs den 13 juli 2016.

Utrustning	**Modell**	**Pris per styck**	**Antal**	**Totala kostnaden**
Switch	HP 5900AF-48XG-4QSFP+	161 899 SEK [90]	2 stycken	323 798 SEK
Strömadapter	HP A58x0AF 650W	9659 SEK [91]	4 stycken	38 636 SEK
D/A-omvandlare	HP X240 10G SFP+ to SFP+ 3m	95,00 USD (≈ 810 SEK) [92]	42 stycken	≈ 34 020 SEK
UPS (Uninterruptible Power Supply)	HP R7KVA UPS 4U IEC-32A HV Intl Kit	3831,52 AUD (≈ 24 917 SEK) [93]	2 stycken	≈ 49 834 SEK
Basserver	HP DL380 Gen9 12LFF CTO Server	14 421 SEK [94]	6 stycken	≈ 86 526 SEK
Processor 1	HP DL380 Gen9 E5-2640v3 Kit	1980 AUD (≈ 12 872 SEK) [95]	6 stycken	≈ 77 232 SEK
Processor 2	HP DL380 Gen9 E5-2640v3 FIO Kit	881,91 GBP (≈ 9990 SEK) [96]	6 stycken	≈ 59 940 SEK
RAM	HP 16GB 2Rx4 PC4-2133P-R Kit	169,98 GBP (≈ 1925 SEK) [97]	12 stycken	≈ 23 100 SEK
HDD	Seagate Mobile HDD 2048GB 2.5" SATA-600 5400rpm	1099 SEK [98]	72 stycken	79 128 SEK
SSD	SSD-Hårddisk 2.5" Intel S3500 Box 800 GB	5515 SEK [99]	15 stycken	82 725 SEK

Nätverkskort	HP FlexFabric 10Gb 2P 534FLR-SFP+	641 AUD (≈ 4167 SEK) [100]	6 stycken	≈ 25 002 SEK
Lagringskontroller 1	HP H240ar FIO Smart HBA	270 USD (≈ 2301 SEK) [101]	6 stycken	≈ 13 806 SEK
Lagringskontroller 2	HP 12Gb DL380 Gen9 SAS Expander Card	5213 SEK [102]	6 stycken	≈ 31 278 SEK

Bilaga 8: Figuren visar kalkyleringen av elpriset från EON som hämtades den 14 juli 2016 [83].

TRITA 110