Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Ausarbeitung

Object Storage

Proseminar "Datei- und Speichersysteme"

vorgelegt von

Quang-Vinh Martin Tran

Fakultät für Mathematik, Informatik und Naturwissenschaften
Fachbereich Informatik
Arbeitsbereich Wissenschaftliches Rechnen

Studiengang:	Wirtschaftsinformatik
Matrikelnummer:	7028885
Betreuer:	Dr. Michael Kuhn

Hamburg, 2019-02-28

Inhaltsverzeichnis

1 Einleitung

Im heutigen Informationszeitalter - auch als Computer- oder Digitalzeitalter bekannt [1] - ist der Gebrauch von Computern, Smartphones, Tablets und vielen anderen derartigen Geräten selbstverständlich geworden. Es lässt sich zudem beobachten, dass die Anzahl der Nutzer solcher Technologien stetig wächst. Dieses lässt sich durch die sog. Digitalen Revolution begründen, die einen Umbruch durch Digitaltechnik und Computer auslöste und inzwischen in alle Lebensbereiche eingedrungen ist. Mit der rapiden Entwicklung neuer Technologien, wie zum Beispiel digitaler Kommunikationssysteme, entstand auch die Konnektivität als „Megatrend unserer Zeit“ [2]. Von Jahr zu Jahr steigt die Nutzung von Geräten, die Daten erzeugen, womit sich auch die Erwartungen an elektronischen Dienstleistungen sowie die damit verbundenen Technologien erhöhen. Nutzer möchten überall und zu jeder Zeit mit dem Internet über soziale Netzwerke wie z. B. Facebook, Twitter oder ähnliche Plattformen verbunden sein und z. B. über WhatsApp oder Snapchat kommunizieren.

Auch auf beruflicher Ebene hat sich viel verändert: Neue Arbeitsmodelle, z. B. Arbeiten von zuhause (Homeoffice) werden von Unternehmen längst akzeptiert, benötigen allerdings auch technische Voraussetzungen, um über geographische Grenzen hinweg eine Form von Kommunikation zu bieten. Das Internet und World Wide Web selbst ermöglicht neue Wege der Monetarisierung; so entstanden neue Industriezweige wie das E-Business. Echtzeit-Datenanalysen sollen hier z. B. auf den (potenziellen) Kunden zugeschnittene Angebote kreieren, damit Güter jeglicher Art von Nutzer über Plattformen wie Amazon oder eBay erworben werden.

1.1 Wohin führt uns dieser Trend?

Es handelt sich um anhaltende Trends, die eine gewaltige Zahl an übertragenen und generierten Daten mit sich bringen: Jeden Tag wird eine Menge von 2.500 Petabyte an Daten erzeugt. Laut der US-amerikanischen Zeitschrift Forbes wurden allein in den vergangenen 2 Jahren 90% aller Daten der Welt generiert; man spricht inzwischen von Big Data. Der Verarbeitung und Speicherung dieser gewaltig wachsenden Datenmassen gerecht zu werden, wird immer mehr zum Problem. „Aus diesem Grund ist die Nachfrage – durch das Management und die Vertreter der IT-Abteilungen – nach Speicherlösungen, die in der Lage sind, noch mehr digitale Inhalte für lange Zeit zu verwalten und zu archivieren, drastisch gestiegen.“ (Michael Nuncic, Kroll Ontrack in „Die Weiterentwicklung des Speichers: Dateispeicher vs. Blockspeicher vs. Objektspeicher – Teil 1“ [3]). Das Unternehmen Kroll Ontrack erläutert, dass der damit entstandene Bedarf jedoch nicht bloß in hardwaretechnischer Dimension, also einer größeren Menge an Speichergeräten, besteht, sondern auch den Bedarf an geeigneten Dateisystemen selbst, welche das Wachstum der Big Data bewältigen können, meint. Firmen wie Microsoft oder Amazon stellten Cloud-Dienste wie Amazon S3 oder Microsoft Azure vor, die jedoch mit der alten Speicherung - wie Datei- und Blockspeicher - nicht mehr kompatibel sind [3]. Eine Lösung ist die sog. Object Storage bzw. der objektbasierte Speicher. Aber inwiefern unterscheidet Letzterer sich von den anderen Speicherkonzepten und löst dieser das Problem der Verarbeitung und Speicherung unseres rasanten Datenzuwachses?

Im weiteren soll diese schriftliche Ausarbeitung die Entwicklung zum Objektspeicher sowie sein Konzept näher betrachten und dabei auch auf die formulierte Fragestellung aufgreifen, um anschließend ein Urteil bezüglich der Eignung zu treffen.

2 Datei- und Speichersysteme sowie deren Konzepte

2.1 Kurzer Einblick in Dateisysteme

Das Dateisystem ist in der Regel ein fester Bestandteil von Betriebssystemen und beschreibt eine Ablageorganisation auf einem Datenträger eines Computers. Neben der Aufgabe, Dateien strukturiert in Verzeichnissen anzulegen [4], soll das Ordnungs- und Zugriffssystem zudem Daten und Metadaten verwalten [5]. Mit dem technologischen Fortschritt und neu gestellten Anforderungen genügte die erste Form von Dateisystem, das lineare Dateisystem in Form von Lochkarten oder -bändern, zur Bewältigung obig genannter Anforderungen nicht mehr; es folgte das Konzept des hierarchischen Speichersystems mit Über- und Unterverzeichnissen. Architekturen zu einem verteilten Dateisystem, einem Netzwerkdateisystem, ließen nicht lange auf sich warten. Anwendern ist es nun möglich, Filesharing zu betreiben und auf Daten zuzugreifen, die auf entfernten Rechnern liegen[6].

2.2 Netzwerkspeichersysteme - Entwicklung zum Objektspeicher hin

Für Dateien, die lokal auf einem Computer gespeichert wurden, aber von allen Rechnern des Netzwerks erreichbar sein sollten, wurden auch neue Formen von Speichersystemen eingeführt. Im kommenden Abschnitt wird flüchtig die NAS- und SAN-Architektur näher in Augenschein genommen werden. Auf Grundlage Letzterer sollen dazugehörige technische Umsetzungen und gleichzeitig die Vorgänger des Objektspeichers, der Datei- und der Blockspeicher, beleuchtet werden.

2.2.1 NAS-Speichersystem

Das Speicherkonzept Network Attached Storage (NAS) baut auf vorhandenen Netzstrukturen auf und basiert auf zentralisierter Datenhaltung, um Dateien auf Entfernung zur Verfügung zu stellen [7]. Die Architektur soll hier von Speichergeräten abstrahieren [8].
NAS dient hierbei ausschließlich dem Zwecke des Speicherns von Daten und besteht aus einem sog. NAS-Server, welcher an das Unternehmensnetz angeschlossen ist. Über Letzteres erfolgt der gesamte Datenverkehr. Das Speicherkonzept löst Speicherprobleme normaler, aber nicht „überdimensionierter“ Netzproportionen. File-Sharing wird bei Network Attached Storage zwischen verschiedenen Betriebssystemen ermöglicht, die Speicherlösung nutzt hierbei die vorhandene Hardware sowohl als Speicher als auch als Netzwerkkomponenten. Insgesamt lässt sich NAS plattformübergreifend einsetzen. Zudem ist die Implementation wenig kompliziert. Die technische Umsetzung von NAS erfolgt über den sog. Datei-Speicher (engl. File Storage/File-Level-Storage/File-Based-Storage). Unter Verwendung von Protokollen wie NFS (Network File System) oder dem CIFS (Common Internet File System) kann der Datenzugriff erfolgen[9]. Aus diesem Grund sind Benutzerrechte, Dateisperren und andere Sicherheitsmaßnahmen zu verwalten.

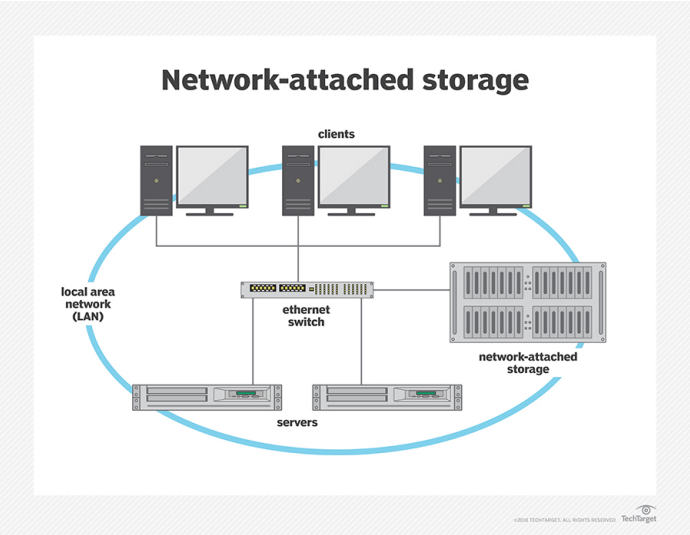

Abbildung 2.1: NAS-Architektur

https://cdn.ttgtmedia.com/rms/onlineImages/network_attached_storage.jpg [letzter Zugriff: 09.01.2019 13:17 Uhr]

2.2.2 SAN-Speichersystem

Im Gegensatz zum NAS-Speichersystem, welches wie bereits erläutert, ein vorhandenes Unternehmensnetzwerk nutzt, bezeichnet ein Storage Area Network (kurz SAN) ein Speichernetz, welches Server und Speichersystem über Breitbandnetze miteinander verbindet und gegenseitig entkoppelt [10]. Die Architektur ermöglicht einen leichten Zugriff von unterschiedlichen Servern aus und lässt sich von beliebigen parallel verteilten Dateisystemen nutzen. Die Datenspeicherung erfolgt in Form von Block-Storage, welches Konzept im weiteren Verlauf näher erläutert wird. Für einen Block-Zugriff auf Daten kommen das Fibre-Channel-Protokoll oder auch iSCSI zum Einsatz [7].

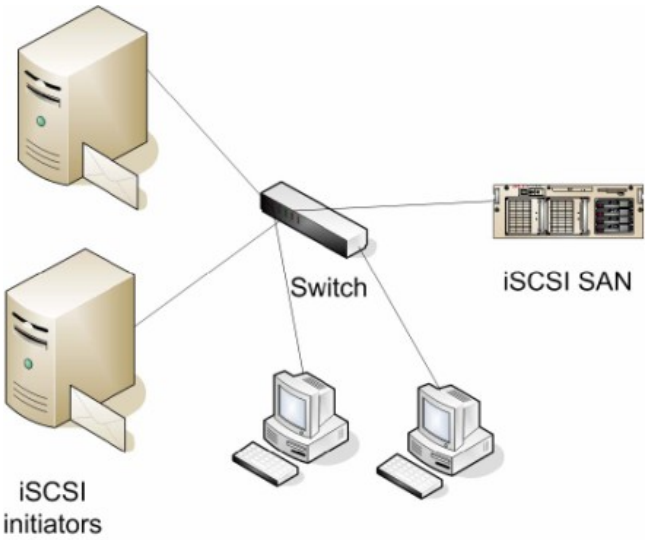

Abbildung 2.2: SAN-Architektur

https://cdn.ttgtmedia.com/digitalguide/images/Misc/xtips_iscsi_2.jpg [letzter Zugriff: 09.01.2019 13:17 Uhr]

2.3 Datei- und Blockspeicher

Wie im Voraus erläutert, handelt es sich bei Dateispeichern (File Storage) und Blockspeichern (Block Storage) um Methoden, Daten auf NAS-Speichersystem- und SAN-Speichersystem-Systemen zu speichern. Es soll nun ein Vergleich beider Systeme vollzogen werden. Es ist anzumerken, dass kommende Abschnitte bzgl. Datei- und Blockspeicher auf Quelle [11] beruhen.

2.3.1 Dateispeicher

Data Storage zeichnet sich durch seine Form des Dateizugriffs aus: Dieser erfolgt über die Adressierung des Dateinamens. Der Host, der die Datei gespeichert hat, arbeitet nach Erhalt des Lese- oder Schreibbefehls bzgl. einer kompletten Datei per Blockzugriff den jeweiligen Datenträger ab. Dadurch, dass der Dateiname für Datenabrufe oder -manipulationen genügt und die physische Adresse - also der Speicherort der Datei - nicht bekannt sein muss, ist ein gleichzeitiger Zugriff auf einen einzigen Datenbestand möglich. Allerdings nimmt man hierbei in Kauf, dass bei Nutzung von NAS-Systemen (in Form von Dateispeichern) Zugriffe auf ihre Dateien nur mit Hilfe eines separaten Rechners (Fileserver) zu bewältigen sind. Von Nachteil ist zudem, dass eine höhere Komplexität bzgl. genannter Operationen sowie eine möglicherweise eingeschränkte Leistungsfähigkeit erkauft wird.

2.3.2 Blockspeicher

Diese Art von Speicher hat seinen Namen von seiner Art des Informationszugriffs: Es handelt sich um ein „Block für Block“-Abarbeiten auf dem Datenträger, welches das Dateisystem auf dem zugreifenden Rechner steuert. Damit wird im Gegensatz zum Dateispeicher kein weiterer Rechner für Lese- o. Schreibzugriffe benötigt. Blockzugriffe sind hiermit deutlich schneller als die Methoden des Dateispeichers, jedoch beschränkt sich jede Operation auf den jeweiligen Rechner allein; parallele Zugriffe sind damit problematisch.

2.4 Zwischenfazit

Die Ansprüche bzgl. der Nutzung, Speicherung und ubiquitären Verfügbarkeit von Daten ist - wie schon erläutert - größer denn je. Vorherig genannte Technologien sind jedoch nicht in der Lage, den Erwartungen bzw. Anforderungen gerecht zu werden; es werden zunehmend leistungsfähige, verteilte und vor allem leicht skalierbare Datenspeicher benötigt, welche vorzugsweise kostengünstig sein sollen.

3 Ansätze zur Bewältigung von Big Data

Der in der Einleitung bereits kurz aufgegriffene Begriff „Big Data“ ist inzwischen vielfältig in verschiedensten Bereichen und Branchen definiert worden; u.a. meint der Begriff nicht nur die bloße Ansammlung von Datenmengen, sondern auch damit verbundene Technologien, die große Massen verwalten und analysieren können. In diesem Kapitel sollen Lösungen für diese Problematik vorgestellt werden.

3.1 „Scale-Up“- vs „Scale-Out“-Speicher

Nachfolgend soll ein E-Paper der Universität Hannover herangezogen werden, um die Begrifflichkeit der Skalierbarkeit zu erläutern [12]. Oliver B. Kaul beschreibt im Rahmen des Software Engineerings die Skalierbarkeit als Fähigkeit eines Systems, sich wachsenden Ansprüchen an die Leistungsfähigkeit anzupassen. Kaul zufolge werden Systeme sowohl auf Hardware- als auch auf Softwareebene skaliert. Skalieren, welches letztendlich der Leistungssteigerung von Rechnernetzwerken, verteilten Dateisystemen etc. dient, unterscheidet zwischen horizontaler (scale up) und vertikaler Skalierbarkeit (scale in): Vertikale Skalierbarkeit bezeichnet das Hinzufügen von Ressourcen innerhalb einer logischen Einheit (eines Rechnerknotens), bspw. Hinzufügen zusätzlicher CPUs zu einem Server.
Horizontale Skalierbarkeit bedeutet, dass man ganze, logische Einheiten hinzufügt. Damit ist man - im Gegensatz zum horizontalen Skalieren - nicht an hardwaretechnische Grenzen gebunden; theoretisch kann also beliebig hoch skaliert werden. Eine Umsetzung eines solchen „Scale Out File Systems“ ist beispielsweise das Hadoop Distributed File System (HDFS). IBM zufolge wird HDFS verwendet, um einen einzelnen Rechnerverbund (Cluster) auf hunderte (oder sogar tausende) Knoten zu skalieren. Es handelt sich um ein verteiltes Dateisystem, das große Datenmengen verarbeitet, die auf handelsüblicher Hardware ausgeführt werden [13].
Das Scale-up-Konzept als solches löst zwar das geschilderte Problem von skalierbaren Speicherkonzepten/Dateisystemen, allerdings wird das Aufrechterhalten als arbeitsintensiver Prozess beschrieben: Der Hersteller von Festplattenlaufwerken und Speicherprodukten Western Digital (WD) umschreibt Scale-Out-File-Systeme als komplexe Systeme verbunden mit hohen Kosten: Nicht nur die nötige dauerhafte Wartung, sondern auch der Datenschutz sind äußerst kostspielig; WD zufolge werden standardmäßig Daten 3-fach redundant gespeichert [14].

3.2 Cloud Computing und Public Cloud

Das Recherchieren über Cloud-Technologien aus neutralen, verlässlichen Quellen hat sich als äußerst schwierig gestaltet, da zumeist Informationen zu Cloud Computing von eigenen Websites von Unternehmen bereitgestellt werden, um ihre Produkte/Dienstleistungen zu bewerben. Nachfolgend werden daher Cloud-Lösungen und Dienstleistungen von IBM - welches weltweit als eines der führenden IT- und Beratungsunternehmen - als Beispiel herangezogen:

Cloud Computing, oft einfach als „Cloud“ bezeichnet, ist die Bereitstellung von On-Demand-Computing-Ressourcen - von Anwendungen bis hin zu Rechenzentren - über das Internet. Zur Verfügung stellende Unternehmen verkaufen ihre Technologie unter dem Pay-for-Use-Konzept und bewerben Cloud Computing mit besonders leichter Skalierbarkeit. Sucht man nach Cloud Computing-Dienstleistungen werden i.d.R. „Software as a service“ (Saas), „Platform as a service“ (Paas), „Infrastructure as a service“ (Iaas) und inzwischen auch „Storage as a service“ angeboten. Bei Inanspruchnahme solcher Dienstleistungen erhält man Zugang zu den auf „fernen“ Computern bereitgestellte Software, Plattformen für z.B. webbasierte Anwendungen oder eben die auch gesamte Infrastruktur (Server, Netzwerke, Datenspeicher,

Software etc.) [15].

Zur Reduktion auftretender Komplexität verteilter Dateisysteme wie bspw. bei HDFS nutzen Unternehmen Public Clouds, welche bspw. von IBM als Systeme mit elastischen Ressourcen und schneller und einfacher Skalierbarkeit zur Befriedigung der eigenen Nachfrage propagiert werden. Das pay-as-you-go-Zahlungsmodell, das mit diesen angebotenen Dienstleistungen gekoppelt ist, ist für Unternehmen mit kleineren Datenbeständen von Vorteil; es kann je nach Bedarf eine Anpassung vorgenommen werden. Gleichwohl sprengen die Kosten für größere Datenmengen den Rahmen[14].

4 Der Objektspeicher

Die mögliche Rettung für vorherig genannte Probleme könnte Object Storage mit on-premise (server-basierter) Technologie darbieten; sie dient ebenfalls der Umsetzung von Cloud-Technologien. In einer kosteneffektiven, hochskalierbaren Umgebung ermöglicht der Objektspeicher Serviceanbietern - wie bspw. IBM oder WD - hunderte von Petabytes in einem einzigen Namensraum (nähere Erläutungen diesbezüglich im kommenden Unterkapitel) ohne jeglichen Leistungsabfall zu skalieren. Dieses Kapitel soll sich nachfolgend mit dem Konzept, der Funktionsweise und Vorteilen von Objektspeichern befassen, um anschließend die Architektur zu beleuchten.

4.1 Konzept „Objektspeicher'

4.1.1 Objekt - Definition

Daten können herkömmlicherweise über Namensräume, also einer baumartigen Anordnungsform und entsprechende Pfadnamen, eindeutig identifiziert werden [16]. Anders ist dies bei Object Storage, welche - wie typischerweise bei Verwendung einer Cloud - eine „hierarchisch freie" Methode zur Datenspeicherung nutzt. Es handelt sich um diskrete Dateneinheiten, sog. Objekte, die auf der gleichen (und einzigen) Hierarchieebene gespeichert werden. Damit Anwendungen gezielt auf bestimmte Daten / Dateien zugreifen können, verfügt jedes Objekt neben Daten über eine einzigartige, darüber eindeutig identifizierbare Kennung sowie zusätzliche Metadaten"[17].
In objektbasierten Speichern wie bspw. „Google Cloud" sind Objekte in sog. Buckets aufbewahrt. Diese sind Container, die die Objekte enthalten. Mithilfe der Buckets lassen sich Daten organisieren und der Zugriff auf die Daten steuern. Allerdings gibt es aufgrund des hierarchiefreien Konzepts entsprechend keine verschachtelten Buckets - anders als bei Verzeichnissen und Ordnern in bspw. File Storage [18].

4.1.2 Metadaten: Organisation unstrukturierter Daten

Metadaten sind strukturierte Daten/Informationen über Merkmale anderer Daten. Objektmetadaten bieten somit Informationen über einzelne Objekte; bei Computerdateien werden diese Metadaten in Form von Dateiattributen gespeichert [19]. Die Verwendung dieser Metadaten als "weitere, leistungsstarke Funktion"(O'Connell, Mark [20]) erleichtern Nutzern das Finden, das Benutzen sowie den Erhalt und eine Wiederverwendung von (insb. unstrukturierten) Daten [21]. Jedes Objekt ist fest mit seinen zugehörigen Metadaten verknüpft und wird mit diesen gespeichert. Die Aufbewahrung der Objektmetadaten in einer separaten, zentralisierten Datenbank, die dem Object Storage Data Services Architekten O'Connell (DELL EMC) zufolge schnell unhandliche Größen erreichen könnte, ist so nicht vonnöten [20].
Die Menge an Daten, besonders in Form von z.B. Textdokumenten, Bildern/Fotos oder gar Audio- und Videodateien nimmt von Tag zu Tag zu. Es handelt sich hierbei um sog. unstrukturierte Daten, welche als Informationen einer nicht identifizierbaren und nicht normalisierten Datenstruktur vorliegen [22]. Diese individuellen Informationselemente sind nicht in einer strukturierten Datenbank organisiert und eignen sich daher nicht für Speicherkonzepte wie den Blockspeicher [20].
Da i.d.R. eine textuelle Suchen von Nutzern erfolgt, kann mit Hilfe von Metadaten eine Einschränkung der Suchergebnisse erfolgen. Im Vergleich zum Dateispeicher, bei dem oftmals alleinig der Dateiname als einzige Auskunft über mögliche Inhalte vorhanden ist, kann man bei einem Objektspeicher die Suche nach seinen Objekten präzisieren: Analog zu bspw. einem Regal verfügt der Object Store über ausführliche und klare Beschreibungen bezüglich Name, Hersteller/Autor etc. O'Connell führt zu Metadaten weitere Vorteile auf: So ermöglichen diese „Daten über Daten" anwendungs- und bedarfsorientierte Anpassungen bezüglich der Anforderungen von Speichernutzern und Anwendungen.

Im Zusammenhang mit Datenanalysen gewinnen Unternehmen durch angehängte Metadaten einen Einblick in bspw. das Nutzerverhalten, um neue Erkenntnisse über deren Präferenzen zu erlangen [20].

4.1.3 Object Storage - ein Speichersystem ohne Hierarchie

Wie bereits grob erläutert, unterscheidet sich ein Objektspeicher dahingehend von einem Dateispeicher, als dass dieser keine Speicherhierarchie besitzt. Das Unternehmen DELL definiert das Konzept und seine Vorteile in seinem Glossar [23] wie folgt: Eine Speicherstruktur ohne jegliche Hierarchieebenen bedeutet, dass ein Objektspeicher keinen Verzeichnisbaum mit (Über-)Ordnern und Unterordnern besitzt. Stattdessen spricht man von einer eindeutigen, globalen Kennung (Identifikationsnummer).
Dies ist von Vorteil, da Anwendungen sowie dessen Anwender den Speicherort der Datei nicht kennen müssen; bei gewünschtem Abruf der Daten wird lediglich die Kennung eingegeben, sodass das System diese ausgibt. Damit ist eine Standortunabhängigkeit möglich - ein Dateisystem, welches die Platzierung von Daten regelt, ist damit überflüssig. Obwohl es sich um ein verteiltes Speichersystem handelt, erscheinen die verschiedenen Rechnerknoten und Standorte als ein einziges, logisches System.
Anhand von sog. Speicherrichtlinien (engl. storage policies) wird die Segmentierung des Objektspeichers innerhalb eines einzelnen Rechnerverbunds für verschiedene Anwendungsfälle ermöglicht. So werden einzeln von Objekt zu Objekt die Replikation (dazu im kommenden Kapitel mehr) und die geographische Verteilung von Daten anhand dieser Policies durchgeführt; DELL zufolge existiert keine dedizierte Replikations- und Backupinfrastruktur.

5 Systemarchitektur „Objektspeicher“ und weitere Funktionen

Nachdem das Konzept von object storage devices (engl., kurz OSD) vorgestellt wurde, soll sich das nächste Kapitel näher mit der Systemarchitektur beschäftigen. Dabei beruhen die Informationen bzgl. der Systemarchitektur größtenteils auf der von Herrn Kanatorn Jindarak sowie Putchong Uthayopas veröffentlichten Arbeit, die im Rahmen der „IEEE 18th International Conference on Parallel and Distributed Systems“ im Jahr 2012 veröffentlicht wurde [24]. Neben der Objektreplikation existieren jedoch noch weitere Funktionen, die das Konzept OSD mit sich bringt. Bevor die Systemarchitektur thematisch behandelt wird, sollen auch diese Kernfunktionen/-eigenschaften, orientiert an der wissenschaftlichen Arbeit von S. Samundiswary and Nilma M. Dongre abgehandelt werden; diese publizierten ihre Arbeit anlässlich der ICISC 2017 [25],

5.1 Weitere Kernfunktionen und Eigenschaften des Objektspeichers

5.1.1 REST API

Cloud - und somit auch in diese Systeme integrierte Objektspeicher - verwenden Schnittstellen zur Programmierung von Anwendungen. Man spricht im englischen Raum vom „Application Programming Interface“, kurz APIs. Um auf Anwendungen zuzugreifen, verwendet Cloud häufig APIs wie REST (Representational State Transfer) sowie SOAP (Simple Object Access Protocol). Aufgrund ihrer zustandslosen Architektur gelten diese als weit verbreitet; man verbindet unter deren Namen häufig „requests for service“ übers Internet. Cloud-Provider-API untersützen sowohl HTTTP- als auch HTTP(s)-Protokoll und verwenden Operationen wie GET, PUT und DELETE zum Erstellen, Aktualisieren, Abrufen sowie Löschen von Daten.

5.1.2 Datenschutz

Bei Objektspeichern handelt es sich um sehr beanspruchbare bzw. strapazierbare Speicher. Jedes Speicherobjekt verfügt über einen Hashwert oder eine Objekt-ID, worüber sich Duplikate lokalisieren lassen, die verworfen werden sollen. OSD arbeitet mit einer redundanten Speicherung auf vielen Geräten in mehreren Einrichtungen einer Region. Zuzüglich zur Verschlüsselung wird der Datenschutz in objektbasierten Speichern durch drei weitere implementierte Mechanismen garantiert: Die Replikation, die Versionierung sowie dem Löschcodierungsmechanismus „Erasure Coding“. Object Storage verwendet das Konzept „Continuous Data Protection“ (CDP), oft auch Continuous Backup oder Real Time Backup genannt. Dieses Verfahren dient fortlaufender, inkrementeller Datensicherung. CDP dient dem Schutz von Datenbeständen vor Effekten korrumpierter Daten, welche erst nach einiger Zeit nach erfolgter, nachteiliger Manipulation festgestellt werden. Eine Wiederherstellung wird vom CDP-Verfahren zu beliebigen, nicht vorab definierten Zeitpunkten ermöglicht [26]. Dabei ist Continuous Data Protection äußerst kosteneffektiv in Speichersystemen anzuwenden. Das CDP-Framework kombiniert Cloud-Objektspeicher und Caching auf effiziente Art und Weise, um Anforderungen von niedrigen Kosten, hoher Kapazität, niedriger Latenz und hohem Speicherdurchsatz gerecht zu werden.

5.1.3 Erasure Coding

Michael Nuncic beschreibt in einem Artikel für Kroll Ontrack [27] das Problem von RAID für Objektspeicher wie folgt:
Die RAID-Technologie soll den Nutzer vor einem Datenverlust aufgrund eines Hardwarefehlers von einer oder mehrerer defekter Festplatten schützen und funktioniert mit kleineren Datenmengen, welche

in den letzten Jahren üblich waren, gut. Allerdings stellen größere Datenbestände das Konzept der Paritätsberechnung und Datenwiederherstellung vor Probleme; so kann eine Wiederherstellung einer vollständigen Festplatte mit 10 Terabyte oder mehr Monate dauern. Aus diesem Grund wird die Erasure-Codierungstechnologie, die methodisch einer klassischen RAID-Absicherung/-Wiederherstellung ähnelt, verwendet. Ein System mit Erasure Coding (EC) teilt jedes Datenelement (Objekt) in kleine Fragmente auf. Nuncic zufolge seien diese Datenblöcke typischerweise mehrere Megabyte groß und daher auch viel größer als die Blöcke, die normalerweise in einem RAID-geschützten System erstellt werden. „Jeder Datenblock wird dann analysiert und zusätzlich zu dem ursprünglichen Datenblock werden mehrere kleinere Fragmente erzeugt." Bei EC wird jedes Fragment codiert und anschließend an vielen Orten auf beliegen Speichermedien gespeichert. Dies ist von Vorteil, da ein Fehler auf einem Laufwerk dann nicht zu Datenverlusten führt. Verglichen zu RAID ist ein Wiederherstellungsprozess deutlich schneller, da nicht alle Teile der ursprünglichen Daten, sondern nur eine bestimmte Anzahl der Fragmente, wiederhergestellt werden müssen. Möchte man seine Originaldaten wiederherstellen, wird eine Mindestmenge dieser sog. Scherben (Blockteile, engl. Shards) benötigt. Zur Fragmentierung wird ein XOR-Scheduling-Algorithmus als spezielle mathematische Formel herangezogen; auf diese soll in dieser Ausarbeitung aber nicht weiter eingegangen werden. Erasure Coding eignet sich für große Datenmengen und alle Anwendungen bzw. Systeme mit geforderter, hoher Ausfalltoleranz - daher auch verteilte Speicheranwendungen und Objektspeicher [25]. Anders als bei RAID ist es zudem deutlich einfacher, Fragmente an anderen Speicherorten zu speichern, als ein vollständiges Backup auf einem externen Speichersystem zu schaffen.

5.1.4 Objektversionierung

Die Funktion Objektversionierung dient dem Abrufen bereits gelöschter oder überschriebener Objekte. Der IT-Riese Google beschreibt mit seiner Cloud-Lösung „Google Cloud" in seiner Dokumentation dazu [28] object versioning (engl.) wie folgt vor:
Für Buckets kann ein Nutzer die Objektversionierung aktivieren. Nach der Aktivierung wird jedes Mal, wenn die Liveversion eines Objekts überschrieben oder gelöscht wird, eine archivierte Version des Speicherobjekts vom Speichersystem (z.B. Google Cloud Storage) generiert. Auch die archivierte Version besitzt wie die Liveversion den gleichen Objektnamen; um diese jedoch voneinander unterscheiden und zugleich auch das Archivobjekt identifizieren zu können, wird jedes Objekt mit einer Generierungsnummer versehen.
Ist die Funktionalität aktiviert, so ist der Anwender vor dem Überschreiben oder versehentlichem Löschen seiner Daten geschützt, d.h. archivierte Versionen eines Objekts lassen sich auflisten, die Liveversion eines Objekts in einen älteren Zustand wiederherstellen. Zudem lassen sich Archivversionen endgültig löschen. Ein Bucket, bei dem die Objektversionierung deaktiviert wurde, sammelt keine neuen archivierten Versionen mehr. Allerdings bleiben vorhandene Objektversionen bestehen.

Näheres zur Objektversionierung

Im Falle von Umsetzungen wie der Google Cloud Storage werden zwei Eigenschaften eines Speicherobjekts verwendet, um die Version eines Objekts festzustellen. Der Dokumentation von Google Cloud [28] zufolge handelt es sich einerseits um die Version der Objektdaten und andererseits um die Version der Objektmetadaten. Nun noch mehr zu Generierungsnummern eines Objekts: Meta-Generierungsnummern kennzeichnen Objekte auch auf Meta-Ebene. Eine Objektgenerierungsnummer wird, unabhängig davon, ob die Objektversionierung aktiviert ist, genutzt, damit Nutzer „Datenressourcen eindeutig identifizieren und Vorbedingungen für die Unteilbarkeit mehrstufiger Transaktionen festlegen" [29]; im Falle von z.B. parallelen Uploads, wobei Objekte in mehrere Teile aufgesplittet und letztlich wieder zusammengesetzt werden, kann so vermieden werden, dass falsche Komponenten zusammengeraten.

Archiv- vs. Liveobjekte

Archivierte Objekte besitzen eigene Metadaten zwecks Unterscheidung von dem jeweiligen Liveobjekt. Außerdem ist es wichtig, dass die archivierten Versionen ihre Zugriffssteuerungslisten im Sinne einer Identitäts- und Zugriffsverwaltung behalten und diese nicht unbedingt über dieselben Berechtigungen wie die Liveversion des Speicherobjekts verfügen. Eine Zugriffssteuerungsliste enthält in der Regel Informationen bezüglich der Berechtigung und dem Bereich [30]. Es wird also festgelegt, welche Aktionen ausgeführt werden können (z.B. Lesen o. Schreiben) und wer die angegebenen Operationen ausführen darf. Manchmal bezeichnet man den Bereich auch als Empfänger.

5.1.5 Objektreplikate

In der Vergangenheit wurden große Speichersysteme mithilfe großer Einzelplattenspeicher, welche mit einem Storage Server verbunden waren, verwirklicht. Verwendet man mehrere Plattengeräte zusammen, lassen sich Zuverlässigkeit und Skalierbarkeit verbessern. Das Konzept RAID ermöglicht es, größere und schnellere Speicher kostengünstig zur Verfügung zu haben. Jedoch - wie schon erläutert - ist die Wiederherstellung von Daten zeitaufwendig.
Bei objektbasierten Speichersystemen gilt die Replikation von Daten/Speicherobjekten im Kontext der Erhöhung von Zuverlässigkeit und Verfügbarkeit von Datenzugriffen als unverzichtbar. Bei diesem Ansatz wird eine Kopie eines Speicherobjekts, ein Speicherobjekt-Replikat, erzeugt und über das objektbasierte Speichersystem verteilt. Dadurch verringert sich die Last bei Zugriffen für alle Speicherserver im Speichersystem; allerdings wird der für jedes Objekt verwendete Speicherverbrauch erhöht.

6 Object-Storage-Lösungen

Dieses Kapitel befasst sich mit zwei verschiedenen Realisierungen, die beispielhaft das Konzept des Objektspeichers auf verschiedene Art und Weise umsetzen.

6.1 Objektspeicher in Cloud Storage Systemen

Das Konzept von OSD wird zunehmend zur Verwirklichung von Cloudsystemen verwendet. Wie Cloud Storage mit Hilfe von Objektspeichern im Zusammenspiel mit anderen Cloud-Systemkomponenten arbeitet, soll in diesem Teil unter die Lupe genommen werden.
Aus abstrakter Sicht besteht i.d.R. ein Cloud Storage System aus 3 Komponenten; dem Client (Benutzer, oft in Form einer Anwendung o. eines Anwendungsservers), dem Manager-Server und dem Object Store selbst. Um auf das Speicherobjekt vom Client- oder Anwendungsserver aus zuzugreifen, sind drei Schritte erforderlich:

1. Zunächst sendet der Anwendungsserver ein „request“ (engl. für Anfrage), um auf ein Speicherobjekt zuzugreifen, an den Manager-Server.

2. Anschließend antwortet der Manager-Server (engl. „response") mit den Objektstandort- und Berechtigungsnachweisdaten im Sinne eines sicheren Zugriffs des Anwendungsservers auf das Objekt.

3. Als Letztes verwendet der Anwendungsserver die Standort- und Berechtigungsdaten, um auf das Speicherobjekt in einem Speicherserver zuzugreifen, der am OSD beteiligt ist.

6.1.1 Cloud-Systemarchitektur - der Manager Server

Wie der vorherige Abschnitt klarstellt, erfolgen Zugriffe auf Objekte über APIs, welche mit dem Manager-Server kommunizieren. Die Architektur des Manager-Servers, fügt sich aus 6 Komponenten zusammen:

A. Activity Collector
Alle Anfragen (requests) von Seiten der Applikationsserver werden gesammelt. Diese „request activities“ bestehen aus dem Namensraum des Speicherobjekts sowie Informationen über die „bandwith consumption“, sprich der benötigten Datenrate bezüglich des requests. Letztere wird benötigt, um entsprechende Planungen der Zuordnung und Platzierung des Speicherobjekts zu arrangieren.

B. Replication Manager
Dieses Modul nutzt Daten des Activity Collectors und zusätzlich Statusdaten zur Analyse und Erstellung des Replikations-Datasets. Das Zugriffsmuster des Anwendungsservers wird zur Erstellung des Replikationsplans verwendet.

C. Garbage Collector
Der Garbage Collector überwacht die Anforderungsaktivitäten und Metadaten des Speicherobjekts, um festzustellen, welche Objekte gelöscht werden können. Jedes Replikationsobjekt besitzt eine Lebensdauer, die regelmäßig aktualisiert wird. Die Objektmetadaten werden vom Garbage Collector aktualisiert. Dieser markiert ebenfalls die Objekte, deren Lebensdauer abgelaufen ist, als zu entfernendes Objekt. Bei Verwendung dieser Technik wird ein Objektspeicherplatz zurückgefordert, wenn die Häufigkeit der Zugriffsanfragen gering ist.

D. Storage Object Controller
Diese Operationseinheit kommuniziert direkt mit OSD. Das Modul erzeugt und entfernt Replikate von Speicherobjekten. Nachdem eine Operation am OSD ausgeführt wurde, wird der Storage Object Indexer benachrichtigt, entsprechende Objektmetadaten zu aktualisieren.

E. Storage Object Indexer
Wie in D. erläutert, ist diese Komponente für das Aktualisieren der Objektmetadaten der Speicherobjekte zuständig. Die Metadaten beziehen sich auf die Speicherobjektmetadaten, den Speicherort seiner Replikate, die Anzahl gleichzeitiger requests sowie die Lebensdauer des Speicherobjekt-Replikats. Dazu kommt, dass der Storage Object Indexer den Speicherort der Replikationsobjekte an Applikationsserver sendet, während er Speicherobjekt-Replikate, die als zu entfernen markiert wurden, nicht mit übersendet.

F. Task Scheduler
Dieser kontrolliert den Replikationszeitraum, indem das Modul benachrichtigt wird, ein Objekt zu replizieren.

6.2 Ceph (RADOS)

Eine softwaretechische Umsetzung, die ein hochskalierbares Speichersystem, basierend auf Objektspeichern, anbietet, ist Ceph. In einem Artikel vom Javamagazin von Herrn Schneller und Dr. Pustina [31] wird das Ceph-Storage-Cluster wie folgt beschrieben:
Ceph ist ein hochperformanter Object Store und verteiltes Datensystem. Im Gegensatz zur bereits vorgestellten möglichen Umsetzung von Cloud Storage arbeitet Ceph nicht mit einer Form von Manager-Server, sondern nutzt die „Intelligenz“ aller beteiligten Rechnerknoten (Object Stores). Damit wird die Notwendigkeit einer zentralen Verwaltungsinstanz vermieden. Sein Kernbestandteil ist ein sogenannter Reliable Autonomic Distributed Object Storage (Kurzform RADOS). Mithilfe von RADOS können alle anfallenden Aufgaben nahezu beliebig horizontal verteilt werden. Die Daten werden nahezu dynamisch verteilt und bei Bedarf möglichst schnell und effizient reorganisiert. Zudem ist ein Zugriff zu jeder Zeit gewährt. Dem Artikel zufolge wird auf dem jeweiligen Server für jedes OSD ein einzelner Ceph-OSD-Daemon-Prozess gestartet. Seine zentrale Aufgabe ist es, ein konkretes Stück Storage-Hardware in möglichst effizienter Art und Weise bezüglich Zugriffszeiten zu verwalten und gleichzeitig Datensicherheit im Cluster zu gewähren. Die Ceph-Technologie wirbt auf seiner eigenen Website damit, dass es mit seiner Software Library den Nutzern Anwendungen anbietet, einen direkten Zugriff auf die das objektbasierte RADOS-System zu erlangen [32]. Das Object Storage Interface „RADOS Gateway“ ermöglicht außerdem eine Zugriff auf kompatible Objektspeicher, wie z.B. Amazon S3 oder OpenStack Swift.

6.2.1 „Ceph Storage Platform“-Architektur

Ceph bietet als Storage Platform - anders als herkömmliche Objektspeichersysteme - verschiedene Sichten bzw. Interfaces sowie Implementationsmöglichkeiten für die Verwendung eines Objektspeichers mithilfe der Ceph-Software. Dabei wird die Ceph Storage Platform architektonisch in 5 Bestandteile/-Sichten unterteilt:

A. RADOS
Wie bereits erläutert, basiert die Softwarelösung Ceph auf einem Objektspeichersystem, dem RADOS.

B. LIBRADOS
Ceph bietet seine eigene Bibliothek mit Frameworks zur Implementation über verschiedenste Apps an und unterstützt hier die Programmiersprachen C, C++, Java, Python, Ruby und PHP.

C. RADOS Gateway
Das RADOS GW fungiert hier als Schnittstelle für Applikationen, um direkten Zugriff auf RADOS zu haben und verschiedene Operationen am Objektspeicher durchzuführen. Dieses Interface ist, wie

bereits erläutert, zugriffskompatibel mit Objektspeichern wie Swift von Openstack oder auch S3 von den Amazon Web Services.

D. RADOS Block Device
Obwohl es sich um einen Objektspeicher handelt, bietet Ceph eine Blockspeicher-Sicht an. Operationen werden über dieses Interface auf herkömmliche Blockspeicherweise ausgeführt und die Distribution der einzelnen Blöcke sowie das Speichern und Manipulieren der Daten wird intern im RADOS geregelt, während über das Block-Device-Interface zugreifende Rechner eine Sicht kreiert wird, als würde dieser mit einem normalen Blockspeicher arbeiten.

E. Ceph File System
Zusätzlich zur Blockspeichersicht gewährt das Ceph File System als Interface eine Dateispeichersicht auf das RADOS. Analog zum Block Device erfolgen aus Sicht eines auf das Ceph FS zugreifenden Rechners Operationen wie auf Dateispeichern.

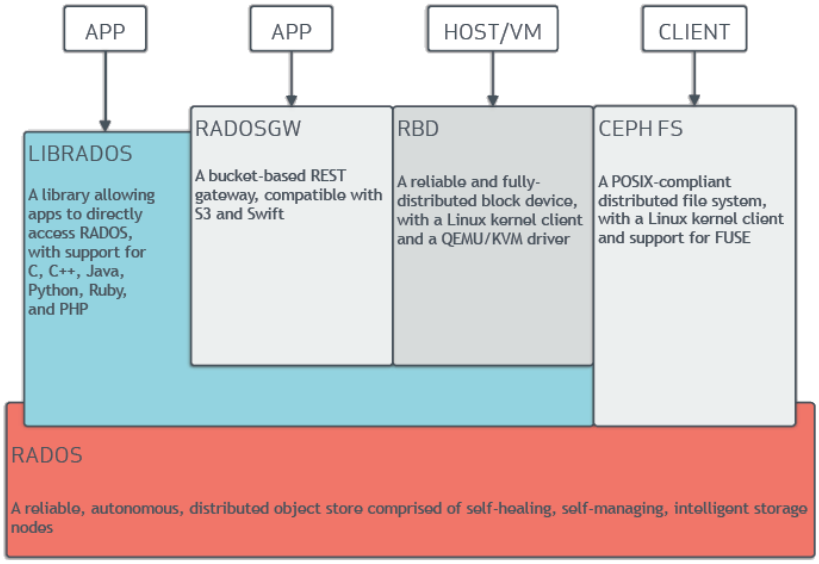

Abbildung 6.1: Architekturdiagramm, das die Komponentenbeziehung der Ceph-Speicherplattform zeigt

`https://raw.githubusercontent.com/ceph/ceph/master/doc/images/stack.png`
`[letzter Zugriff: 12.01.2018 07:54 Uhr]`

7 Zusammenfassung und Fazit

Zu guter Letzt soll die anfangs aufgestellte Frage aufgegriffen und beantwortet werden:
Inwiefern eignen sich Objektspeicher, um das Problem der Datenverwaltung und -speicherung großer Massen, auch in Zukunft zu lösen?

Beim Objektspeicher handelt es sich um ein verteiltes Speichersystem, welches hochskalierbar ist. Das Einbinden von Metadaten macht verglichen zu seinen Vorgängern den Unterschied aus und ermöglicht eine leichte Datenverwaltung und Speicherstandortunabhängigkeit über direktes Ansprechen mit Hilfe von Objekt-IDs. Parallele Zugriffe auf Daten, geografische Grenzen und hoher Rechenaufwand durch Suchalgorithmen sind nichtig. Technologien wie Erasure Coding, Objektversionierung und Objektreplikaten garantieren zudem eine hohe Datensicherheit und einfache Datenwiederherstellung. Bei vielen Realisierungen fallen außerdem Kosten für zusätzliche Rechnerknoten für die Zugriffsverwaltung und Datenplatzierung weg, denn in der Regel erfolgt ein Zugriff über eine REST-API.

Kosten, die an einer Stelle wegfallen, können sich beim Objektspeicher jedoch an anderer Stelle wieder erhöhen, denn die dreifache Objektreplikation kann besonders in Exabyte-Dimensionen äußerst kostspielig werden. Es ist bei der Wahl eines (verteilten) Speichersystems also wichtig, abzuwägen, inwiefern sich Kostenreduktion bzw. -erhöhung rentieren. Mit Ausnahme von einigen Objektspeicherformen (wie z.B. Lustre oder Ceph) werden Daten grobgranular gespeichert; in vielen in vielen Object Stores ist alleinig eine Speicherung in Form von (ganzen) Objekten möglich. Ein objektbasierter Speicher eignet sich daher eher für Speicherung von Daten, bei denen in der Regel nur Lese- bzw. Auf-/Abrufoperationen erfolgen. Weniger sinnvoll sind OSDs für Applikationen mit hohen Leistungsanforderungen und einer Vielzahl an Operationen, die die Objekte selbst ändern würden, wie z.B. Transaktions- und Datenbankanwendungen.
Am Ende liegt auch ein Problem der Einheitlichkeit in Definition und Konzeption/Architektur vor: Objektspeicher sind eine verhältnismäßig neue Technologie mit verschiedensten Definitionen, Realisierungen und Auslegungen in der Literatur, die sich teils widersprechen; einen wirklichen Standard scheint es nicht zu geben, auch die Begrifflichkeit des Speicherobjekts wird von Herstellern, Lexika und realisierten Objektspeichern (siehe z.B. Ceph) anders betrachtet. Das „Zusammenfügen“ bzw. Integrieren von zwei oder mehreren verschiedenen Object Storage Systemen könnte sich so als schwierig erweisen. Für Kunden, die erwägen einen Objektspeicher zu nutzen, mag diese Uneinheitlichkeit und fehlende Kompatibilität große Verwirrung hervorrufen und womöglich auch (vorerst) von dieser Speichertechnologie abschrecken.

Ist der Objektspeicher nun die Antwort auf „alles“? Nach Ausarbeitung der Technologie Objektspeicher wird klar, dass dieser mit seiner Vielzahl an Vorteilen eine Lösung ist, die mit zunehmender Ausreifung und Verbreitung ersichtlich an Bedeutsamkeit im Markt gewinnt. Um dem kontinuierlichen Datenwachstum gerecht zu werden, scheint es derzeit so, als seien Unternehmen dazu gezwungen, diese Form von verteilten Speichersystemen zu nutzen. Wie auch erläutert, löst der Objektspeicher sicherlich nicht alle Probleme, doch dank diesem ist die Sorge, Big Data nicht bewältigen zu können, bedeutend geringer. Mit Ausblick auf die Zukunft, in der zusätzlich auftretende Aufwendungen für zunehmende Verwaltung und Speicherung von Daten kaum vermeidbar sein werden, bietet ein Object Store einen guten Kosten-Nutzen-Kompromiss.

Literaturverzeichnis

[1] Wikipedia: *Informationszeitalter*. https://de.wikipedia.org/wiki/Informationszeitalter. [Letzter Zugriff: 23.12.2018 14:57 Uhr].

[2] Zukunftsinstitut: *Megatrend Konnektivität*. https://www.zukunftsinstitut.de/dossier/megatrend-konnektivitaet/. [Letzter Zugriff: 23.12.2018 15:32 Uhr].

[3] Nuncic, Michael, Kroll Ontrack: *Die Weiterentwicklung des Speichers: Dateispeicher vs. Blockspeicher vs. Objektspeicher – Teil 1*. https://www.ontrack.com/de/blog/the-evolution-of-storage-file-storage-vs-block-storage-vs-object-storage-part-1/7865, Februar 2018. [Letzter Zugriff: 23.12.2018 14:57 Uhr].

[4] Wikipedia: *Dateisystem*. https://de.wikipedia.org/wiki/Dateisystem. [Letzter Zugriff: 23.12.2018 16:02 Uhr].

[5] Dr. Kuhn, Michael: *Dateisysteme – Hochleistungs-Ein-/Ausgabe [Powerpoint-Präsentation]*, April 2016.

[6] ITWissen.info: *Netzwerk-Dateisystem*. https://www.itwissen.info/Netzwerk-Dateisystem-network-file-system.html, Mai 2018. [Letzter Zugriff: 20.11.18 18:14 Uhr].

[7] Kerns, Randy (TechTarget): *Der Unterschied zwischen SAN- und NAS-Architektur*. https://www.searchstorage.de/antwort/Der-Unterschied-zwischen-SAN-und-NAS-Architektur, Mai 2016. [letzter Zugriff: 11.12.2018 16:38 Uhr].

[8] Dr. Kuhn, Michael: *Parallele verteilte Dateisysteme – Hochleistungs-Ein-/Ausgabe [Powerpoint-Präsentation]*, April 2016.

[9] Rouse, Margaret (TechTarget): *File Storage*. https://www.searchstorage.de/definition/File-Storage, April 2016. [letzter Zugriff: 23.12.2018 17:10 Uhr].

[10] ITWissen.info: *Speichernetz*. https://www.itwissen.info/Speichernetz-storage-area-network-SAN.html, Mai 2018. [letzter Zugriff: 11.12.2018 16:55 Uhr].

[11] Froehlich, Karl: *Grundlagen: Network Attached Storage*. https://www.tecchannel.de/a/grundlagen-network-attached-storage,402522,3, April 2005. [letzter Zugriff: 11.12.2018 18:12].

[12] Kaul, Oliver B.: *Kurz & gut: Skalierbarkeit [PDF]*. http://www.se.uni-hannover.de/pub/File/kurz-und-gut/ws2011-labor-restlab/RESTLab-Skalierbarkeit-Oliver-Beren-Kaul-kurz-und-gut.pdf, November 2011. [Letzter Zugriff auf im Internet bereitgestellte PDF: 27.12.2018 15:02 Uhr].

[13] IBM: *HDFS*. https://www.ibm.com/analytics/hadoop/hdfs. [letzter Zugriff: 27.12.2018 15:07 Uhr].

[14] McWhorter, Mike: *Why Object Storage? A Short, Definitive Explanation*, März 2018. [letzter Zugriff: 27.12.2018 15:16 Uhr].

[15] IBM: *What is cloud computing?* https://www.ibm.com/cloud/learn/what-is-cloud-computing. [letzter Zugriff: 27.12.2018 15:31 Uhr].

[16] Wikipedia: *Namensraum*. https://de.wikipedia.org/wiki/Namensraum, September 2017. [letzter Zugriff: 27.12.2018 16:01 Uhr].

[17] IBM: *What is object storage?* https://www.ibm.com/cloud/learn/what-is-object-storage [letzter Zugriff: 27.12.2018 16:27 Uhr].

[18] Google-Cloud: *Wichtige Begriffe - Buckets.* https://cloud.google.com/storage/docs/key-terms, September 2018. [letzter Zugriff: 30.12.2018 19:30 Uhr].

[19] Wikipedia: *Metadaten.* https://de.wikipedia.org/wiki/Metadaten, September 2018. [Letzter Zugriff: 27.12.2018 16:54 Uhr].

[20] O'Connell, Mark: *Was ist Objektspeicher? Entdecken Sie Cloud- und Softwarebasierten Speicher für Ihr Rechenzentrum.* https://germany.emc.com/storage/elastic-cloud-storage/articles/what-is-object-storage-cloud-ecs.htm. [letzter Zugriff: 28.12.2018 14:40 Uhr].

[21] UNC-Libraries: *Metadata for Data Management: A Tutorial.* https://guides.lib.unc.edu/c.php?g=8749&p=44500, Juli 2017. [letzter Zugriff: 27.12.2018 17:00 Uhr].

[22] Litzel, Nico: *Was sind unstrukturierte Daten?* https://www.bigdata-insider.de/was-sind-unstrukturierte-daten-a-666378/, November 2017. [letzter Zugriff: 27.12.2018 17:08 Uhr].

[23] EMC, DELL: *EMC Glossar - Objektspeicher.* https://germany.emc.com/corporate/glossary/object-storage.html. [letzter Zugriff: 28.12.2018 14:03 Uhr].

[24] Jindarak, Kanatorn und Uthayopas Putchong: *Enhancing Cloud Object Storage Performance using Dynamic Replication Approach.* Technischer Bericht, HPCNC, Department of Computer Engineering, Faculty of Engineering, Kasetsart University, 2012.

[25] Samundiswary, S. und Nilma M. Dongre: *Object Storage Architecture in Cloud Storage for Unstructured Data (ICISC 2017).* techreport, Department of Information Technology, RAIT, Nerul, Navi Mumbai, 2017.

[26] Wikipedia: *Continuous Data Protection.* [letzter Zugriff 12.01.2019 17:31 Uhr].

[27] Nuncic, Michael: *Die Weiterentwicklung des Speichers: Dateispeicher vs. Blockspeicher vs. Objektspeicher – Teil 2.* https://www.ontrack.com/de/blog/the-evolution-of-storage-file-storage-vs-block-storage-vs-object-storage-part-2/7880, Februar 2018. [letzter Zugriff: 06.01.2018 12:07 Uhr].

[28] Google-Cloud: *Objektversionierung.* https://cloud.google.com/storage/docs/object-versioning?hl=de, Dezember 2018. [letzter Zugriff: 05.01.2019 12:47].

[29] Google-Cloud: *Generierungsnummern und Vorbedingungen.* https://cloud.google.com/storage/docs/generations-preconditions?hl=de, Dezember 2018. [letzter Zugriff: 05.01.2019].

[30] Google-Cloud: *Zugriffssteuerungslisten (ACLs).* https://cloud.google.com/storage/docs/access-control/lists?hl=de, Dezember 2018. [letzter Zugriff: 05.01.2019].

[31] Schneller, Daniel und Lukas Dr. Pustina: *Ceph Storage Cluster.* Javamagazin, 2015.

[32] Ceph: *Object Storage.* https://ceph.com/ceph-storage/object-storage/. [letzter Zugriff: 11.01.2019 13:42 Uhr].

[33] ITWissen.info: *NAS (network attached storage).* https://www.itwissen.info/NAS-network-attached-storage-NAS-Speicher.html, Juni 2017. [letzter Zugriff: 23.12.2018 17:10 Uhr].

[34] Rouse, Margaret (TechTarget): *Storage Area Network (SAN).* https://www.searchstorage.de/definition/Storage-Area-Network-SAN, Oktober 2007. [letzter Zugriff: 24.12.2018 11:27 Uhr].

[35] Rouse, Margaret (TechTarget): *Block Storage*. https://www.itwissen.info/Speichernetz-storage-area-network-SAN.html, Oktober 2014. [letzter Zugriff: 11.12.2018 18:12 Uhr].

[36] Center, NetApp Documentation: *Object versioning*. https://docs.netapp.com/sgws-110/index.jsp?topic=Januar 2018. [letzter Zugriff: 02.01.2019 17:55 Uhr].

Appendices

Abbildungsverzeichnis