Symcloud: Filesync and Collaboration platform

Wachter Johannes

Inhaltsverzeichnis

1 Einleitung **5**

1.1 Projektbeschreibung 5

1.2 Inspiration 6

1.3 Technologie 6

1.3.1 Sulu CMF 8

1.3.2 Symfony2 8

1.3.3 PHPCR 8

1.3.4 PHP 8

1.4 Anforderungen 8

2 Stand der Technik **8**

2.1 Verteilte Systeme 8

2.2 Dropbox 8

2.3 ownCloud 10

2.4 Diaspora 11

2.5 Zusammenfassung 11

3 Cloud Computing **11**

4 Speicherverwaltung **11**

4.1 Datenhaltung in Cloud-Infrastrukturen 12

4.2 Amazon Simple Storage Service (S3) 12

4.2.1 Versionierung 14

4.2.2 Skalierbarkeit 14

4.2.3 Datenschutz 14

4.2.4 Alternativen zu Amazon S3 15

4.2.5 Performance 15

4.3 Verteilte Dateisysteme 16

4.3.1 NFS 17

4.3.2 Ceph 17

4.3.3 Sheepdog 17

4.3.4 GlusterFS 17

4.3.5 XtreemFS 17

4.3.6 Speichergeschwindigkeit 17

4.4 Datenbank gestützte Dateiverwaltungen 17

4.4.1 MongoDB & GridFS 17

4.4.2 Crate 17

4.5 Performance 17

4.6 Evaluation 17

Anhang **17**

Abbildungsverzeichnis

1 Überblick über die Komponenten 7

2 Blockdiagram der Dropbox Services (Quelle https://www.dropbox.com/help/1968) 9

3 ownCloud Enterprise Architektur Übersicht (Quelle ownCloud 2015) 10

4 Bereitstellungs Szenario von ownCloud(Quelle ownCloud 2015) . 11

5 Versionierungs Schema Amazon S3 (Quelle „Using Versioning“ 2015) 14

6 Upload Analyse zwischen EC2 und S3 (Quelle „Amazon S3 and EC2 Performance Report – How fast is S3¿‘ 2009) 16

Tabellenverzeichnis

1 Objekt Metadaten 13

1 Einleitung

Seit den Abhörskandalen durch die NSA und anderen Geheimdiensten ist es immer mehr Menschen wichtig, die Kontrolle über die eigenen Daten zu behalten. Aufgrund dessen erregen Projekte wie Diaspora[1], ownCloud[2] und ähnliche Software Lösungen immer mehr Aufmerksamkeit. Die beiden genannten Software Lösungen decken zwei sehr wichtige Bereiche der persönlichen Datenkontrolle ab.

Diaspora ist ein dezentrales soziales Netzwerk. Die Benutzer von diesem Netzwerk sind durch die verteilte Infrastruktur nicht von einem Betreiber abhängig. Es bietet die Mög-lichkeit, seinen Freunden bzw. der Familie, eine private Plattform anzubieten. Das Interes-sante daran ist, dass sich sogenannten Pods (dezentrale Knoten), beliebig untereinander vernetzen lassen und damit ein P2P Netzwerk aufbauen lässt. Pods können von jedem in-stalliert und betrieben werden; dabei kann der Betreiber bestimmen, wer in sein Netzwerk eintreten darf und welche Server mit seinem verbunden sind. Die verbundenen Pods tau-schen ohne einen zentralen Knoten Daten aus und sind dadurch unabhängig. Dies garan-tiert die volle Kontrolle über seine Daten im Netzwerk (siehe „Was ist Dezentralisierung¿‘ 2015).

Das Projekt “ownCloud” ist eine Software, die es ermöglicht, Daten in einer privaten Cloud zu verwalten. Mittels Endgeräte-Clients können die Daten synchronisiert und über die Plattform auch geteilt werden. Insgesamt bietet die Software einen ähnlichen Funktion-sumfang gängiger kommerzieller Lösungen an (siehe „Owncloud Features“ 2015). Zusätzlich bietet es eine Kollaborationsplattform, mit der zum Beispiel Dokumente über einen online Editor, von mehreren Benutzern gleichzeitig, bearbeitet werden können. Diese Technologie basiert auf der JavaScript Library WebODF[3].

1.1 Projektbeschreibung

Symcloud ist eine private Cloud-Software, die es ermöglicht über dezentrale Knoten (ähnlich wie Diaspora) Daten über die Grenzen des eigenen Servers hinweg zu teilen. Verbundene Knoten tauschen über sichere Kanäle Daten aus, die dann über einen Client mit dem Endgerät synchronisiert werden können.

TODO genauere Beschreibung

Die Software baut auf modernen Web-Technologien auf und verwendet als Basis das PHP-Framework Symfony2[4]. Dieses Framework ist eines der beliebtesten in der Open-Source Community. Es bietet neben der Abstraktion von HTTP-Anfragen auch einen Dependency-Injection-Container und viele weitere

[1] https://diasporafoundation.org/
[2] https://owncloud.org/
[3] http://webodf.org/
[4] http://symfony.com/

Komponenten wie zum Beispiel Routing und Event Dispatcher. Zusätzlich erleichtert es die Entwicklung von großen PHP-Projekten, durch die Möglichkeit den Code in Komponenten, sogenannten Bundles, zu gliedern. Diese können dann mit der Community geteilt werden.

Als Basis für die Plattform verwendet Symcloud das Content-Management-Framework SULU[5] der Vorarlberger Firma MASSIVE ART WebServices[6] aus Dornbirn. Es bietet ein erweiterbares Admin-UI, eine Benutzerverwaltung und ein Rechtesystem. Diese Features ermöglichen Symcloud eine schnelle Entwicklung der Oberfläche und deren zugrundeliegenden Services.

1.2 Inspiration

TODO Noch einmal ownCloud - Diaspora und Ted Nelson mit dem Xanadu Projekt

1.3 Technologie

Dieses Kapitel beschreibt die verwendeten Technologie etwas genauer. In Abbildung 1 zeigt die Abhängigkeiten als Schichten Diagramm. Ganz oben ist zum einen die Oberfläche von Symcloud, die in die Sulu Umgebung eingebettet ist. Sulu selbst ist eine "One-Page application", die verschiedene Javascript Komponenten zur Verfügung stellt, um die Anwendung erweitern zu können. Die andere Schnittstelle ganz oben ist der Synchronisierung Client, der es ermöglich Daten über ein Kommandozeilen Programm zu synchronisieren. Beide "Oberflächen" sprechen das Backend über eine gesicherte REST-API an. Diese API wird verwendet um Daten zu empfangen aber auch daten an den Server zu senden. Zum Beispiel sendet der Synchronisierungs Client einen POST-Request an den Server um eine Datei hochzuladen.

Auf der Server-Seite gibt es zum einen die standard API-Schnittstellen von Sulu und zum anderen die Erweiterung durch Symcloud mit der File-API. Als Persistence-Schicht für die Metadaten, der Dateien in der Symcloud, wird die Abstraktionsschicht PHPCR verwendet. Diese Schnittstelle bietet einen Zugriff auf verschiedenste Content-Repositories an.

[5] http://www.sulu.io
[6] http://www.massiveart.com/de

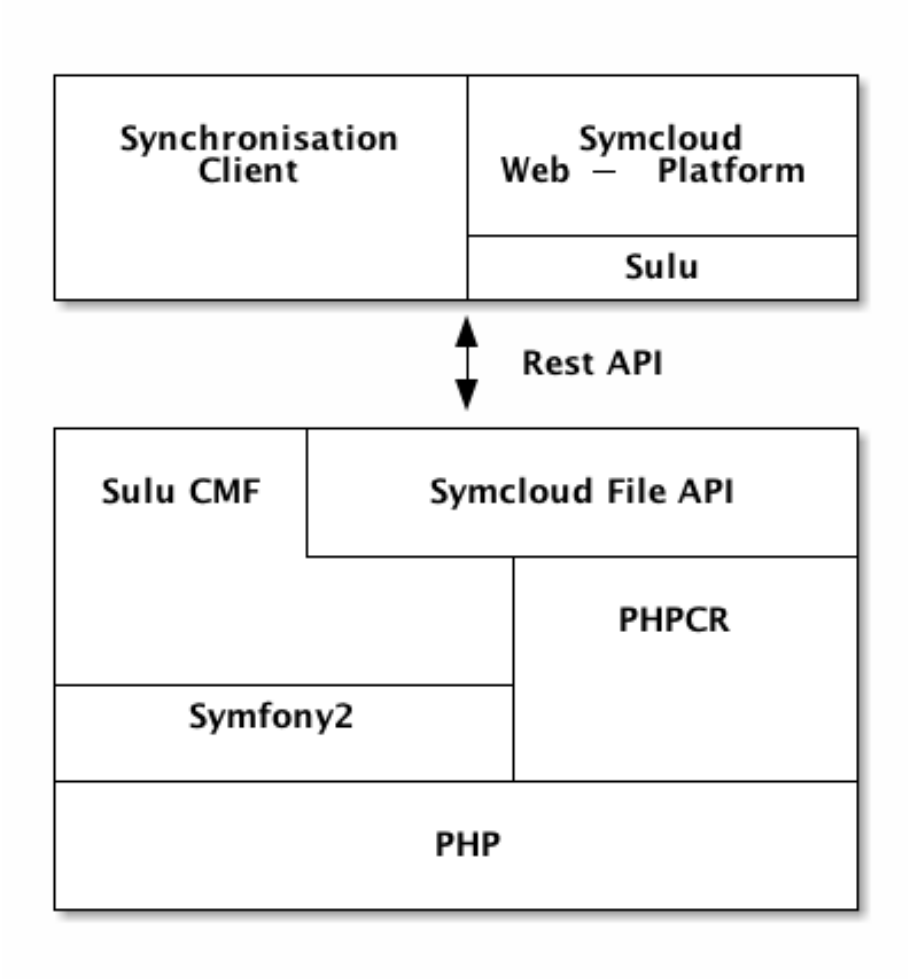

Abbildung 1: Überblick über die Komponenten

1.3.1 Sulu CMF

1.3.2 Symfony2

1.3.3 PHPCR

1.3.4 PHP

1.4 Anforderungen

2 Stand der Technik

In diesem Kapitel werden moderne Anwendungen anhand ihrer Architektur durchläuchtet, um die jeweilige Vorteile für Symcloud zusammenzufassen.

2.1 Verteilte Systeme

Eine Definition von verteilten Systemen gibt Andrew Tanenbaum in seinem Buch Verteilte Systeme:

```
"Ein verteiltes System ist eine Menge voneinander unabhängiger
Computer, die dem Benutzer wie ein einzelnes kohärentes
System erscheinen"
```

Diese Definition beinhaltet zwei Aspekte. Der eine Aspekt besagt, dass die einzelnen Maschinen in einem Verteilten System autonom sind. Der zweite bezieht sich auf die Software, die die Systeme miteinander verbinden. Durch die Software glaubt der Benutzer, dass er es mit einem einzigen System zu tun hat (siehe Tanenbaum ; Steen 2003, p. 18).

Ein nicht mehr ganz neues aber immer noch sehr aktuelles Verteiltes System ist das Netzwerk-Protokoll NFS (Network File Service).

TODO beschreibung eines netzwerk protokolles TODO trennung metadaten und inhalt

2.2 Dropbox

Dropbox-Nutzer können jederzeit von ihrem Desktop aus über das Internet, über mobile Geräte oder über mit Dropbox verbundene Anwendungen auf Dateien und Ordner zugreifen. Alle diese Clients stellen Verbindungen mit sicheren Servern her, über die Sie Zugriff auf Dateien haben, Dateien für andere Nutzer freigeben können, und verknüpfte Geräte aktualisieren können, wenn Dateien

hinzugefügt, verändert oder gelöscht werden. Der Dropbox-Service betreibt verschiedene Dienste, die sowohl für die Handhabung und Verarbeitung von Metadaten als auch von unformatiertem Blockspeicher verantwortlich sind. (siehe „Wie funktioniert der Dropbox-Service¿‘ 2015)

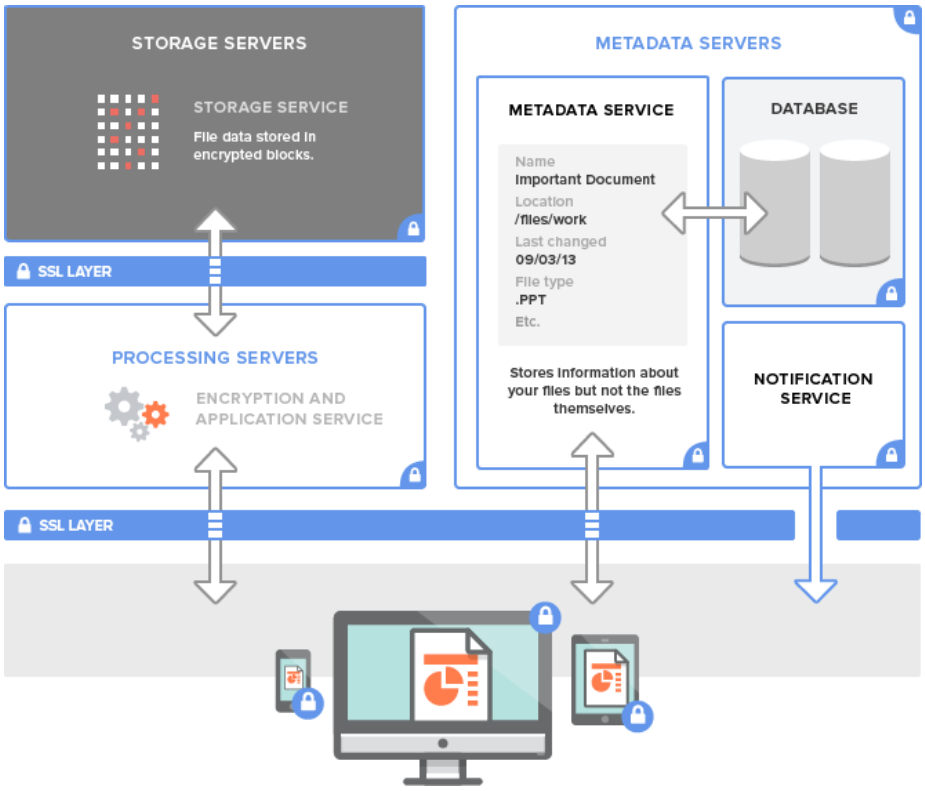

Abbildung 2: Blockdiagram der Dropbox Services (Quelle https://www.dropbox.com/help/1968)

In der Abbildung 2 werden die einzelnen Komponenten in einem Blockdiagram dargestellt. Es gliedert sich in drei größere Blöcke:

- Metadata Servers
- Storage Servers
- Processing Servers

Wie im Kapitel 2.1 beschrieben trennt Dropbox intern die Dateien von ihren Metadaten. Der Metadata Service speichert die Metadaten und informationen zu ihrem Speicherort in einer Datenbank aber der Inhalt der Daten liegt in einem seperaten Storage Service. Dieser Service verteilt die Daten wie ein “Load Balancer” über viele Server.

Der Storage Service ist wiederum von aussen durch einen Application Service abgesichert. Die Authentifizierung erfolgt über das OAuth2 Protokoll (siehe „Core API Dokumentation“ 2015). Diese Authentifizierung wird für alle Services verwendet, also auch für den Metadata Service und der Notification Service.

2.3 ownCloud

Nach den neuesten Entwicklungen arbeitet ownCloud an einem ähnlichen Feature wie Symcloud. Unter dem Namen “Remote shares” wurde in der Version 7 eine Erweiterung in den Core übernohmen, mit dem es möglich ist sogenannte “Shares” mittels einem Link auch in einer anderen Installation einzubinden. Dies ermöglicht es Dateien auch über die Grenzen des Servers hinweg zu teilen. (siehe „Server2Server - Sharing“ 2015)

Die kostenpflichtige Variante von ownCloud geht hier noch einen Schritt weiter. In Abbildung 3 ist zu sehen, wie ownCloud als eine Art Verbindungsschicht zwischen verschiedenen “On-Site”, also Daten die vor Ort bereitstehen, und Daten aus der Cloud dienen soll. (siehe ownCloud 2015, p. 1)

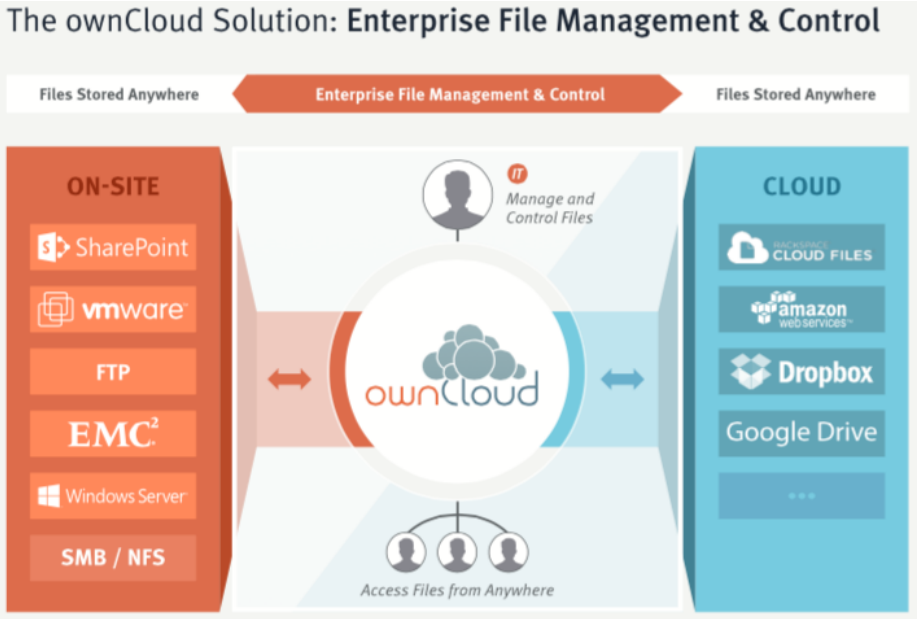

Abbildung 3: ownCloud Enterprise Architektur Übersicht (Quelle ownCloud 2015)

Um die Integration in ein Unternehmen zu erleichtern bietet es verschiedenste Services an. Unter anderem ist es möglich Benutzerdaten über LDAP oder ActiveDirectory zu verwalten und damit ein doppeltes Verwalten der Benutzer zu vermeiden. (siehe ownCloud 2015, p. 2)

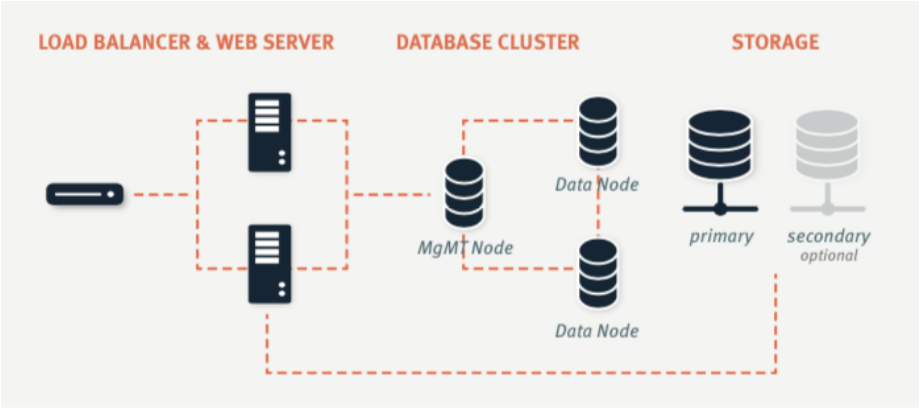

Abbildung 4: Bereitstellungs Szenario von ownCloud(Quelle ownCloud 2015)

Für einen produktiven Einsatz wird eine scalierbare Architektur, wie in Abbildung 4, vorgeschlagen. An erster Stelle, steht ein Load-Balancer, die die Last der Anfragen an mindestens zwei Websterver verteilt. Diese Webserver sind mit einem MySQL-Cluster verbunden, in dem die User-Daten, Anwendungsdaten und Metadaten der Dateien gespeichert sind. Dieser Cluster besteht wiederum aus mindestens 2 Datenbank Servern. Dies ermöglicht auch bei stark frequentierten Installationen eine Horizontale Skalierbarkeit. Dadurch ist der Leistung des gesamten Systems kaum grenzen gesetzt. Zusaätzlich sind die Webserver mit dem File-Storage verbunden. Dieser kann wiederum skaliert werden. Dies wird jedoch als optional bezeichnet. (siehe ownCloud 2015, p. 3-4)

2.4 Diaspora

TODO kaum infos weitere suchen notwendig

2.5 Zusammenfassung

3 Cloud Computing

4 Speicherverwaltung

Ein wichtiger Aspekt von Cloud-Anwendungen ist die Speicherverwaltung. Es bieten sich verschiedenste Möglichkeiten der Datenhaltung in der Cloud an. Dieses Kapitel beschäftigt sich mit der Evaluirung der verschienden Dienste bzw. Lösungen, mit denen Speicher verwaltet und möglichst effizient zur Verfügung gestellt werden können.

Aufgrund der Anforderungen des Projektes werden folgende Anforderungen an die Speicherlösung gestellt.

Ausfallsicherheit Die Speicherlösung ist das Fundament einer jeder Cloud-Anwendung. Ein Ausfall dieser Schicht bedeutet oft ein Ausfall der kompletten Anwendung. ???

Skalierbar Die Datenmengen einer Cloud-Anwendung sind of schwer abzuschätzbar und können sehr große Ausmasse annehmen. Daher ist eine wichtige Anforderung an eine Speicherlösung die Skalierbarkeit. ???

Datenschutz Der Datenschutz ist ein wichtiger Punkt beim betreiben der eigenen Cloud-Anwendung. Meist gibt es eine kommerzielle Konkurenz die mit günstigen Preisen die Anwender anlockt um ihre Daten zu verwerten. Die möglichkeit die Daten privat auf eigenen Server zu speichern sollte also gegeben sein. Damit Systemadministratoren nicht auf einen Provider angewiesen sind.

Flexibilität Um Daten flexibel zu speichern sollte es möglich sein Verlinkungen und Metadaten direkt in der Speicherlösung zu speichern. Dies erleichtert die Implementierung der eigentlichen Anwendung.

Versionierung Ein optionale Eigenschaft ist die Integrierte Versioning der Daten. Dies würde eine Vereinfachung der Anwendungslogik ermöglichen, da Versionen nicht in einem seperaten Speicher abgelegt werden müssen.

Performance ???

4.1 Datenhaltung in Cloud-Infrastrukturen

Wenn man Speicherstrukturen in der Cloud genauer betrachtet gibt es Grundsätzlich drei Möglichkeiten. Zum einen sind dies Objekt Speicherdienste, wie zum Beispiel Amazon S3[7], die es ermöglichen, dass mehrere Instanzen einer Anwendungen gleichzeitig den Speicher verwenden. Eine andere Möglichkeit sind verteilte Dateisysteme oder Datenbank gestützt wie zum Beispiel GridFS[8] von MongoDB.

4.2 Amazon Simple Storage Service (S3)

Amazon Simple Storage Service bietet Entwicklern einen sicheren, beständigen und sehr gut Skalierbaren Objektspeicher an. Es dient der einfachen und sicheren Speicherung großer Datenmengen (siehe „Amazon S3“ 2015). Daten werden in sogenannten Buckets gegliedert. Jeder Bucket kann unbegrenzt Objekte enthalten.

[7] http://aws.amazon.com/de/s3/
[8] http://docs.mongodb.org/manual/core/gridfs/

Die gesamtgröße der Objekte ist jedoch auf 5TB beschränkt. Sie können nicht verschachtelt werden, allerdings können sie Ordner enthalten um die Objekte zu gliedern.

Die Kernfunktionalität des Services besteht darin Daten in sogenannten Objekten zu speichern. Diese Objekte können bis zu 5 GB groß werden. Zusätzlich wird zu jedem Objekt ca. 2kB Metadaten abgelegt. Die Tabelle 1 enthät eine Liste von Systemdefinierten Metadaten. Einige dieser Metadaten können vom Benutzer überschrieben werden, wie zum Beispiel `x-amz-storage-class`, andere werden vom System automatisch gesetzt, wie zum Beispiel `Content-Length` (siehe „Object Key and Metadata“ 2015).

Tabelle 1: Objekt Metadaten

Name	Description
Date	Object creation date.
Content-Length	Object size in bytes.
Last-Modified	Date the object was last modified.
Content-MD5	The base64-encoded 128-bit MD5 digest of the object.
x-amz-server-side-encryption	Indicates whether server-side encryption is enabled for the object, and whether that encryption is from the AWS Key Management Service (SSE-KMS) or from AWS-Managed Encryption (SSE-S3).
x-amz-version-id	Object version. When you enable versioning on a bucket, Amazon S3 assigns a version number to objects added to the bucket.
x-amz-delete-marker	In a bucket that has versioning enabled, this Boolean marker indicates whether the object is a delete marker.
x-amz-storage-class	Storage class used for storing the object.
x-amz-website-redirect-location	Redirects requests for the associated object to another object in the same bucket or an external URL.
x-amz-server-side-encryption-aws-kms-key-id	If the x-amz-server-side-encryption is present and has the value of aws:kms, this indicates the ID of the Key Management Service (KMS) master encryption key that was used for the object.
x-amz-server-side-encryption-customer-algorithm	Indicates whether server-side encryption with customer-provided encryption keys (SSE-C) is enabled.

Zusätzlich zu diesen Systemdefinierten Metadaten ist es möglich Benutzerdefi-

nierte Metadaten zu speichern. Das Format dieser Metadaten entspricht einer Key-Value Liste. Sie sind 2KB limitiert.

4.2.1 Versionierung

Die Speicherlösung bietet eine Versionierung der Dateien. Diese kann über ein Konfigurationsfile in jedem Bucket aktiviert werden.

```
<VersioningConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Status>Enabled</Status>
</VersioningConfiguration>
```

Ist die Versionierung aktiviert gilt diese für alle Objekte, die dieser enthält. Wird anschließend ein Objekt überschrieben, resultiert dies in einer neuen Version, dabei wird die Versions-ID im Metadaten Feld `x-amz-version-id` auf einen neuen Wert gesetzt (siehe „Using Versioning“ 2015). Dies veranschaulicht die Abbildung 5.

Abbildung 5: Versionierungs Schema Amazon S3 (Quelle „Using Versioning“ 2015)

4.2.2 Skalierbarkeit

Die Skalierbarkeit ist aufgrund der, von Amazon, verwalteten Umgebung sehr einfach. Es wird soviel Speicherplatz zur verfügung gestellt wie benötgit wird. Der Umstand, das mehr Speicherplatz benötigt wird, spiegelt sich nur auf der Rechnung des Betreibers ab.

4.2.3 Datenschutz

Amazon ist ein US-Amerikanisches Unternehmen und steht aus diesem, wie andere Dienste, seit Jahren in der Kritik. Um dieses Problem zu kompensieren, können Systemadministratoren sogenannte “Availability Zones” auswählen und

damit steuern wo ihre Daten gespeichert werden. Zum Beispiel werden Daten aus einem Bucket mit der Zone Irland, auch wirklich in Irland gespeichert. Zusätzlich gewährt Amazon die Verschlüsselung der Daten (siehe Wikipedia 2015a).

Wer bedenken hat seine Daten aus den Händen zu geben kann auf verschiedene kompatible Lösungen zurückgreifen.

4.2.4 Alternativen zu Amazon S3

Es gibt einige Amazon S3 kompatible Services die einen ähnlichen Service bieten. Diese sind allerdings meist auch US-Amerikanische Firmen und daher gleich vertrauenswürdig wie Amazon. Wer also auf Nummer sicher gehen will und seine Daten bzw. Rechner-Instanzen ganz bei sich behalten will, kommt um eine Installation von Cluster Lösungen nicht herum.

Eucalyptus Eucalyptus ist eine Open-Source-Infrastruktur zur Nutzung von Cloud-Computing auf Rechner Cluster. Der Name ist ein Akronym für "Elastic Utility Computing Architecture for Linking Your Programs To Useful Systems". Die hohe kompatibilität macht diese Software-Lösung zu einer optimalen Alternative zu Amazon-Web-Services (siehe Wikipedia 2015b).

Riak Cloud Storage ???

4.2.5 Performance

HostedFTP veröffentlichte im Jahre 2009 in einem Perfomance Report ihre Erfahrungen mit der Performance zwischen EC2 (Rechner Instancen) und S3 (siehe „Amazon S3 and EC2 Performance Report – How fast is S3¿‘ 2009). Über ein Performance Model wurde festgestellt, das die Zeit für den Download einer Datei in zwei Bereiche aufgeteilt werden kann.

Feste Transaktionszeit Eine fixe Zeiteinheit, die für die bereitstellung order erstellung der Datei benötigt wird. Beeinflusst wird diese Zeit kaum, allerdings kann es aufgrund schwankender Auslastung zu Verzögerungen kommen.

Downloadzeit Ist liniar abhängig zu der Dateigröße und kann aufgrund der Bandbreite schwanken.

Ausgehend von diesen Überlegungen kann davon ausgegangen werden, dass die Upload- bzw. Downloadzeit einen linearen Verlauf über die Dateigröße aufweist. Diese These wird von den Daten unterstützt. Aus dem Diagram (Abbildung 6) kann die feste Transaktionszeit von ca. 140ms abgelesen werden.

Für den Download von Dateien ensteht laut den Daten aus dem Report keine fixe Transaktionszeit. Die Zeit für den Download ist also nur von der Größe der Datei und der Bandbreite anbhängig.

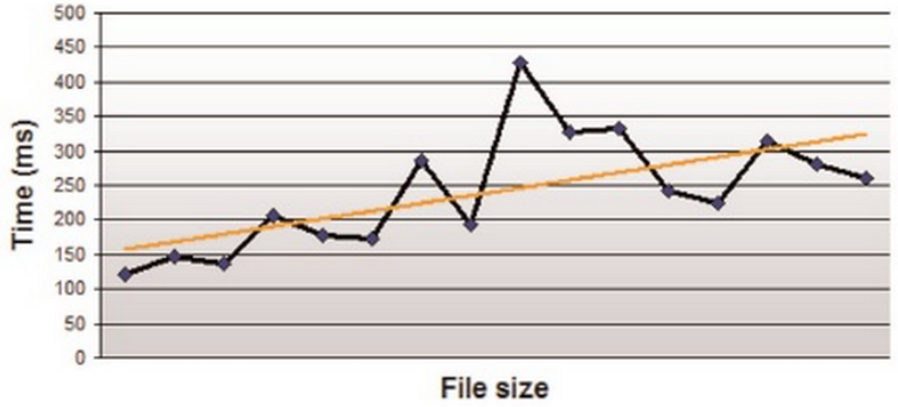

Abbildung 6: Upload Analyse zwischen EC2 und S3 (Quelle „Amazon S3 and EC2 Performance Report – How fast is S3¿‘ 2009)

4.3 Verteilte Dateisysteme

- http://member.wide.ad.jp/~shima/publications/20120924-dfs-performance.pdf

4.3.1 NFS

4.3.2 Ceph

4.3.3 Sheepdog

4.3.4 GlusterFS

4.3.5 XtreemFS

4.3.6 Speichergeschwindigkeit

4.4 Datenbank gestützte Dateiverwaltungen

4.4.1 MongoDB & GridFS

4.4.2 Crate

4.5 Performance

4.6 Evaluation

Anhang

„Amazon S3 and EC2 Performance Report – How fast is S3¿‘ (2009): Amazon S3 and EC2 Performance Report – How fast is S3? https://hostedftp.wordpress.com/2009/03/02/.

„Amazon S3“ (2015): Amazon S3. . Online im Internet: http://aws.amazon.com/de/s3/ (Zugriff am: 08.04.2015).

„Core API Dokumentation“ (2015): Core API Dokumentation. . Online im Internet: https://www.dropbox.com/developers/core/docs (Zugriff am: 26.03.2015).

„Object Key and Metadata“ (2015): Object Key and Metadata. . Online im Internet: http://docs.aws.amazon.com/AmazonS3/latest/dev/UsingMetadata.html (Zugriff am: 08.04.2015).

ownCloud (2015): ownCloud Architecture Overview . Online im Internet: https://owncloud.com/de/owncloud-architecture-overview

„Owncloud Features“ (2015): Owncloud Features. . Online im Internet: https://owncloud.org/features (Zugriff am: 05.03.2015).

„Server2Server - Sharing“ (2015): Server2Server - Sharing. . Online im Internet: https://www.bitblokes.de/2014/07/server-2-server-sharing-mit-der-owncloud-7-schritt-fuer-schritt (Zugriff am: 27.03.2015).

Tanenbaum, A.S. ; Steen, M. van (2003): Verteilte Systeme: Grundlagen und Paradigmen. Pearson Education Deutschland GmbH. (= I : Informatik). Online im Internet: https://books.google.at/books?id=qXGnOgAACAAJ

„Using Versioning“ (2015): Using Versioning. . Online im Internet: http://docs.aws.amazon.com/AmazonS3/latest/dev/Versioning.html (Zugriff am: 08.04.2015).

„Was ist Dezentralisierung¿‘ (2015): Was ist Dezentralisierung? . Online im Internet: https://diasporafoundation.org/about (Zugriff am: 05.03.2015).

„Wie funktioniert der Dropbox-Service¿‘ (2015): Wie funktioniert der Dropbox-Service? . Online im Internet: https://www.dropbox.com/help/1968 (Zugriff am: 26.03.2015).

Wikipedia (2015a): Amazon Web Services — Wikipedia, Die freie Enzyklopädie . Online im Internet: http://de.wikipedia.org/w/index.php?title=Amazon_Web_Services&oldid=139854883

Wikipedia (2015b): Eucalyptus (Software) — Wikipedia, Die freie Enzyklopädie . Online im Internet: http://de.wikipedia.org/w/index.php?title=Eucalyptus_(Software)&oldid=137397846