Ausarbeitung Anwendungen 1 - Ws 2011

Sebastian Zagaria

Multicast Backbone in the Cloud

Fakultät Technik und Informatik
Department Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Inhaltsverzeichnis

Kurzzusammenfassung

Heutzutage spielen gruppenkommunikations Anwendungen eine immer wichtigere Rolle im Internet. Eine Herausforderung bei diesen Anwendungen ist es die Daten effektiv zu verteilen. Durch die geringe Verfügbarkeit von IP-Multicast im Internet, werden für gruppenkommunikations Anwendungen alternative Verfahren gesucht und angewendet. Das Thema der vorliegenden Ausarbeitung, ist es ein Softwarekonzept für einen virtualisierten, hoch skalierbaren und selbst konfigurierenden Multicast-Dienst in der Cloud vorzustellen.

1 Einleitung

Heutzutage spielen gruppenkommunikations Anwendungen eine immer wichtigere Rolle im Internet. Beispiele für solche Anwendungen sind, Audio- und Video-Conferencing, Video-Streaming, Datentransfer und multiplayer Games. Eine Herausforderung bei diesen Anwendungen ist es die Daten effektiv innerhalb einer Gruppe von Teilnehmern zu verteilen. Zu diesem Zweck werden Multicast-Verfahren für die Verteilung der Daten verwendet. Allgemein bezeichnet Multicast einen verbindungsloser Übertragungsdienst, welcher Daten an eine Untergruppe von Hosts in einem Netzwerk überträgt. Multicast basiert auf dem Publisher/Subscriber Model. Bei diesem Model existieren ein oder mehrere Publisher, die Daten an eine spezifische Gruppe verteilen möchten. Der Publisher besitzt dabei kein Wissen darüber, welche und ob es überhaupt Empfänger gibt, die Interesse an diesen Daten haben. Ein Subscriber bekundet sein Interesse an den Daten einer spezifischen Gruppe, ohne zu wissen wer der Publisher ist und ob überhaupt einer existiert.

Um den Unterschied zwischen Unicast-Verbindungen und Multicast zu verdeutlichen betrachten wir die Abbildungen 1 und 2. Die Abbildung 1 stellt eine Gruppenkommunikation über Unicast-Verbindungen dar. Das dargestellt Szenario zeigt einen Sender der Daten an eine Menge von Teilnehmern verteilen möchte, die Interesse an diesen Daten haben. An diesem Beispiel lassen sich mehrere Probleme erkennen. Damit der Sender seine Daten an alle Mitglieder der Gruppe senden kann müssen ihm diese bekannt sein, d.h die IP-Adresse oder Namen der Gruppenmitglieder. Der Sender muss diese Daten dann replizieren und iterativ an alle Empfänger senden, wodurch die Paketlaufzeit der Daten deutlich erhöht wird. Die Paketlaufzeit und der Replikationsaufwand steigt mit der Anzahl der Teilnehmer in einer Gruppe.

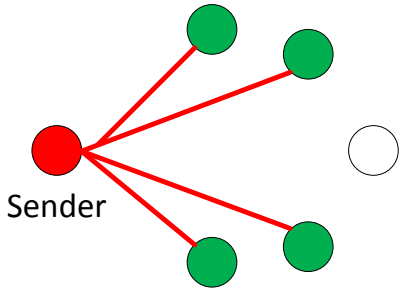

Abbildung 1: Nicht effektive Gruppenkommunikation

Die Abbildung 2 beschreibt ein solches Szenario mit Multicast. Der Sender sendet eine Nachricht an eine Gruppenadresse. Es wird kein Wissen über die Mitglieder der Gruppe Benötigt, alle Teilnehmer werden anhand der Gruppenadresse identifiziert. Nach dem absenden der Nachricht wird diese über Verteiler so häufig repliziert wie benötigt und an die entsprechenden Hosts oder weitere Verteiler gesendet. Bei den Verteiler kann es sich um Endhosts oder Router handeln dies ist abhängig von dem verwendeten Multicast-Algorithmus.

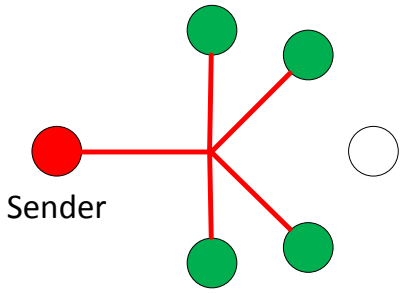

Abbildung 2: Effektive Gruppenkommunikation

Jedoch finden exsistierende Multicastverfahren wie sie duch IPv4 und IPv6 angeboten werden oft keine Verwendung aufgrund ihrer geringen Verfügbarkeit im Internet Diot u. a. (2000). Deshalb verwenden heutige gruppenkommunikations Anwendungen meistens einen auf Anwendungsebene implementierten Multicast-Algorithmus, dieser wird als Application Layer Multicast (ALM) bezeichnet. ALM´s sind nicht so effizient wie die über IP verfügbaren Multicast-Protokolle, aber anders als bei IP bestehen hier keine Probleme bei der Verteilung und Verfüg-

barkeit im Internet. Generell werden ALM´s speziell für eine Anwendung entwickelt und sind nicht portierbar.

Ziel dieser Ausarbeitung ist es ein Konzept für einen Multicast-Dienst vorzustellen der anwendungsunabhängig arbeitet, kostengünstig ist und über eine hoch verfügbare und flexible Infrastruktur verfügt. Als Infrastruktur für einen solchen Multicast-Dienst eignet sich das Cloud-Computing mit seinem flexiblen Bezahlmodel und der gut verteilten Rechenzentren die über große Bandbreite verfügen. Um einen anwendungsunabhängigen Dienst zu realisieren wird eine Multicast-API benötigt die es dem Programmierer ermöglicht gruppenkommunikations Anwendungen unabhängig von dem Multicast-Alogrithmus zu entwickeln. Aus diesem Grund wird der Multicast-Dienst als teil des HAMcast Projektes entwickelt, welches eine solche API zur Verfügung stellt.

Die vorliegende Ausarbeitung gliedert sich wie folgt. Im folgenden Kapitel wird zunächst auf das grundlegende Konzept des Cloud-Computing eingegangen., anschließend wird das HAMcast Projekt vorgestellt. Zum Schluss wird ein Konzept für einen Multicast-Verteilungs-Dienst vorgestellt und abschließend wird eine Zusammenfassung der vorliegenden Ausarbeitung gegeben.

2 Cloud-Computing

Cloud-Computing ist der Ansatz IT-Infrastrukturen, Rechenkapazität und Datenspeicher zu Virtualisieren und bei Bedarf dem Benutzer dynamisch zur Verfügung zu stellen. Der Zugriff auf die Ressourcen der Cloud erfolgt dabei über das Netzwerk/Internet. Die Motivation dahinter ist das Content-Provider nicht mehr in eine eigene IT-Infrastruktur investieren müssen um ihren Kunden einen Service anbieten zu können. Das Risiko einer Überversorgung von Hardwarekapazitäten sowie einer Unterversorgung, wird durch die dynamische Hinzunahmen von Ressourcen innerhalb weniger Minuten, aufgelöst. Die Wartung und der fehlerfreie Betrieb der IT-Infrastruktur innerhalb der Cloud liegt bei den Cloud-Providern.

Das Bezahlmodel beim Cloud-Computing unterscheidet sich ebenfalls von den normalen Mietverträge für Server. Der Benutzer des Cloud-Service zahlt in der Regel abhängig von den verwendeten Ressourcen [Fouquet u. a. (2009)].

- Prozessorlast pro Stunde
- Datenspeicher pro GB
- Datentransfere pro GB

Es gibt mehrere Arten von Services die aus einem solchen Bezahlmodel gegenüber dem herkömmlichem einen Nutzen ziehen können [Armbrust u. a. (2010)].

Erstens, Services deren Auslastung zeitlich abhängig ist. Ein Beispiel dafür wäre ein großes Datencenter dessen Potential aber nur für eine geringe Zeitspanne ausgenutzt wird und dessen Ressourcen den größten Teil der Zeit ungenutzt bleiben. Das Cloud-Computing erlaubt es die Ressourcen für diese geringe Zeitspanne, in der die Lastspitzen auftreten, hinzuzuziehen und wenn diese nicht mehr benötigt werden freizugeben, was potenziell kostensparend ist.

Zweitens, ein Service dessen Auslastung im voraus nicht bekannt ist, z.B IPTV.

- Für eine im voraus hochgelobte Sendung wird über IPTV ausgestrahlt und zu Beginn der Sendung existieren viele Zuschauer für die eine große Anzahl an Ressourcen wird benötigt. Nach einigen Minuten jedoch sinkt die Anzahl der Zuschauer drastisch weil das Interesse an der Sendung stark abnimmt.

2.1 Cloud Architektur

Es gibt derzeit noch keine eindeutige Definition, aber in den meisten fällen wird die Cloud-Architektur in drei Schichten unterteilt Fouquet u. a. (2009).

- **Infrastructure as a Service (IaaS)**
  IaaS bietet einen „low-level“ Service, der den vollen zugriff auf die virtualisierte Hardware gewährt. Der Benutzer kann entscheiden welches Image, z.b Linux oder Windows, auf der virtualisierten Hardware zum Einsatz kommen soll. Load Balancing und die Verwaltung der Software sind dem Benutzer überlassen. Ein beispiel für einen solchen Service IaaS Service ist Amazon EC2.

- **Platform as a Service (PaaS)**
  Der Cloud-Provider bietet eine API an, die von den Anwendungsentwicklern genutzt werden kann um seine Software für die zur Verfügung gestellte Plattform zu entwickeln. Der Benutzer ist dadurch jedoch eingeschränkt bei der Entwicklung der Software, des weiteren besteht nur noch ein begrenzter oder gar kein Zugriff mehr auf die Hardware. Entscheidungen über das Image sowie Load-Balancing werden vom Cloud Provider getroffen. Ein Beispiel für eine solche Plattform ist die Google App Engine.

- **Software as a Service (SaaS)**
  SaaS bezeichnet die Software die in eine Cloud als Infrastruktur nutzt. SaaS Software basiert meist auf eigenen oder durch den Provider vorgegebene PaaS und IaaS. Beispiele für solche Software sind z.B Google Docs, Calendar oder Onlive, eine Spieleplattform in der Cloud die auf einer eigenen Infrastruktur basiert.

2.2 Risiken

Bei dem heutigen Stand des Cloud-Computing besteht ein großes Risiko in dem „single point of failure“ [Armbrust u. a. (2010)]. Sollte ein Cloud-Provider aus irgendwelchen Gründen nicht erreichbar sein oder sogar den betrieb einstellen, ist der Cloud-Service nicht mehr nutzbar und die gespeicherten Daten gehen verloren.

Um diesem Problem vorzubeugen sollte der Service auf verschiedene Cloud-Providern verteilt werden. Zum jetzigem Zeitpunkt hätte ein solches Verfahren jedoch einen hohen Entwicklungsaufwand, da derzeit keine standardisierte API für den Zugriff auf die Cloud existiert. Eine Portierung auf einen anderen Cloud-Provider wäre mit der Implementation einer neuen API verbunden. Für Cloud-Provider mag diese Art von Kundenbindung vorteilhaft erscheinen, jedoch werden dadurch die Möglichkeiten des Cloud-Computing und der Einsatz stark eingeschränkt. Die Schwierigkeiten des Providerwechsel und des „single point of failure“ verhindern, das einige Firmen Cloud-Computing und dessen Vorteile nutzen. Eine Lösung wäre es eine standardisierte API zu entwickeln [Armbrust u. a. (2010)] die es den Cloud-Nutzern ermöglicht ihren Service über mehrere Cloud-Provider anbieten zu können.

3 HAMcast

Multicast existiert in einer Vielzahl von Varianten und Technologien wie z.B. IPv4, IPv6, ASM, SSM und Application-Layer-Multicast. Zurzeit gibt es keine einheitliche Programmierschnittstelle, die den Multicast-Dienst von der Technologie abstrahiert. Somit ist es den Softwareentwicklern überlassen, die eine Gruppenkommunikations-Anwendung implementieren wollen, festzulegen welche Technologie die Anwendung zur Laufzeit benötigt. Daraus ergibt sich, dass heutige Multicast-Anwendungen mit zwei Problemen zu kämpfen haben. Entweder sind sie technologieabhängig und funktionieren nur unter bestimmten Netzwerk-Vorrausetzungen oder es werden Leistungseinbußen in Kauf genommen, um die Anwendung für eine breitere Benutzergruppe zur Verfügung stellen zu können.

Das HAMcast Projekt [Meiling u. a. (2010)] stellt einen universellen Multicast bereit, ohne dabei Veränderungen an vorhandenen Technologien und Protokollen vornehmen zu müssen. Vielmehr werden die bereits vorhandenen Technologien und Protokolle in ihrer Funktionalität durch HAMcast erweitert. Die HAMcast-Architektur bietet Funktionalitäten, die es dem Softwareentwickler ermöglichen Gruppenkommunikations- Anwendungen zu entwickeln und zu implementieren, welche unabhängig von Netzwerktechnologie und Protokollen operieren.

Um einen technologieunabhängigen Dienst zur Verfügung stellen zu können, muss die Adressierung und Namensgebung von der verwendeten Technologie abstrahiert werden. HAMcast

basiert auf dem Identifier-Locator-Split, d.h. Gruppennamen werden von technologieabhängigen Adressen separiert. HAMcast-Anwendungen verwenden Gruppennamen zur Identifikation der Gruppe. Durch diese Namen kann die verwendete Technologie für das Verteilen der Daten vor der Anwendung verborgen werden.

Ein Gruppenname wird in HAMcast durch eine URI repräsentiert. Beispiel für eine URI: *scheme :// group @ instantiation : port / sec-credentials* Das „scheme“ bezeichnet den Namensraum, z.B. ip, sip oder scribe. „group“ gibt die Gruppen ID an. „instantiation“ identifiziert die Entität, die eine Instanz der Gruppe generiert, z.B. bei SSM. „port“ identifiziert eine bestimmte Applikation und sec-credentials ist optional für die Implementierung von Sicherheitsmechanismen. Zulässige Gruppen IDs sind z.B. ip://239.0.0.1:1234, scribe://239.0.0.1:1234 oder sip://snoopy@peanuts.com.

Das Adress-Mapping kann dadurch realisiert werden, dass kleinere IDs in größere Namensräume eingebunden werden oder eine willkürliche unbenutzte Addresse im Zielnamensraum ausgewählt wird. Die Beziehungen zwischen logischen IDs, den Gruppennamen und technischen IDs können durch Mapping Funktionen verwaltet werden. Diese können entweder zustandslos oder zustandsbehaftet sein. Dazu wird ein Late-Binding von technologiespezifischen Adressen und Namen zur Laufzeit vorgenommen. Das Binding von Adressen und Namen zur Laufzeit ermöglicht es Softwareentwicklern, Anwendungen zu implementieren ohne eine vorherige Festlegung auf eine bestimmte Technologie.

HAMcast unterscheidet zwischen zwei Arten von Hosts. Hosts, die sich in einer Multicast-Insel befinden und Hosts, die sich in einem Netzwerk ohne Multicast- Unterstützung befinden. Hosts, die sich in einem nicht multicastfähigem Netzwerk befinden, können sich zu einem Application-Layer-Multicast-Netzwerk über Unicast verbinden.

Multicast-Inseln werden über Interdomain-Multicast-Gateways (IMG) ([Wählisch und Schmidt (2009)], [Wählisch und Schmidt (2007)]) miteinander verbunden. Dazu muss ein IMG Multicastdaten von einer Technologie auf eine andere übersetzen. Um das durchführen zu können, benötigt ein IMG Mechanismen um Gruppennamen zwischen technologiespezifischen Gruppenadressen übersetzen zu können. Informationen über Gruppenzugehörigkeit und Quellen werden dazu benötigt. Die Abbildung 3 zeigt Beispielhaft wie IMG´s die verschiedenen Multicast-Inseln zu einem gemeinsamen Netzwerk verbinden.

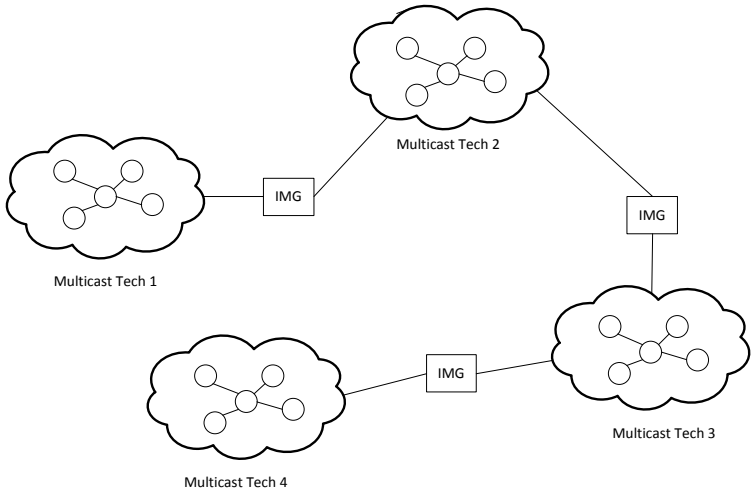

Abbildung 3: Einsatz von IMG´s

Um auf den HAMcast Multicast-Dienst zugreifen zu können, wird als Teil des HAMcast Projektes ein Applikation-Programming-Interface (API) zur Verfügung gestellt, die „common multicast API“ [Wählisch u. a. (2010)]. Die API wird durch eine Middleware realisiert. Die unterstützten Multicast-Technologien sind in Module aufgeteilt. Diese Aufteilung ermöglicht eine hohe Erweiterbarkeit, Anpassung und Flexibilität der Middleware. So können neue Technologien hinzugefügt werden, ohne eine Veränderung der Anwendung. Die Anwendungen kommunizieren über einen Socket-Stub mit der Middleware, der Socket-Stub ist mittels Inter-Prozess-Kommunikation (IPC) mit der Middleware verbunden. Zum Ausführen des HAMcast Multicast-Dienstes muss eine Instanz der Middleware auf jedem Host, der den Dient nutzen möchte, initiiert werden. Die Abbildung 4 zeigt den Aufbau der Middleware.

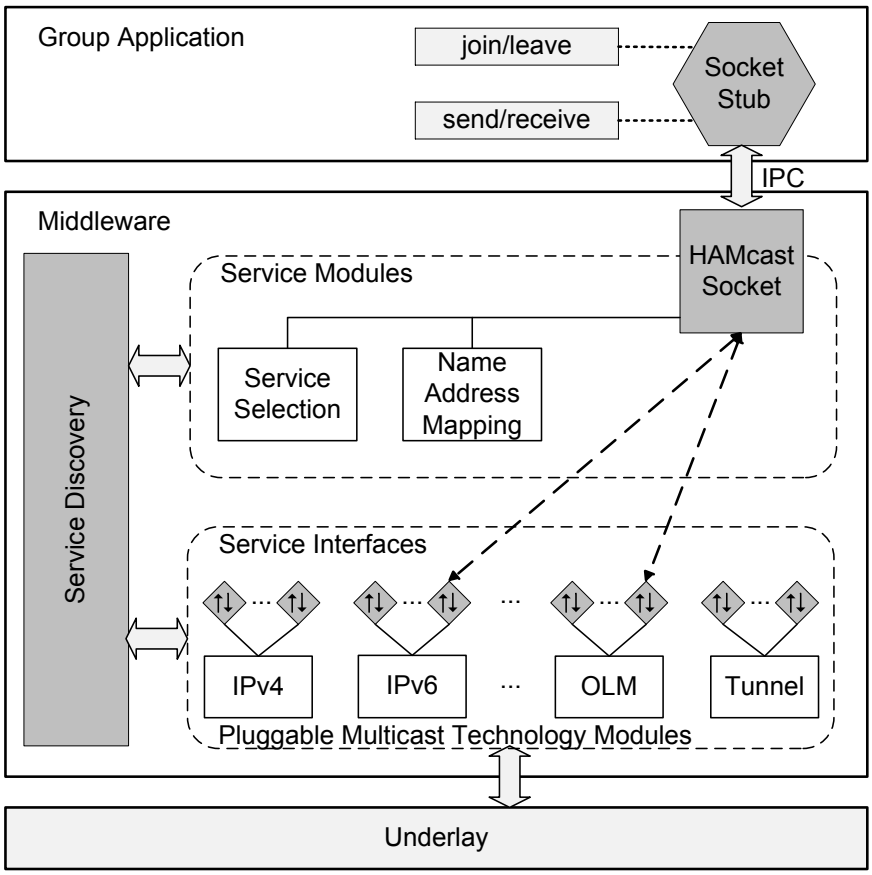

Abbildung 4: Aufbau der HAMcast Middleware

4 Multicast Backbone in the Cloud

4.1 Ziele

Ziel ist die Konzeptionierung eines skalierbaren Multicast-Verteilungsdienst der Daten innerhalb einer Cloud transportiert.

Vier wichtige Kriterien sollen dabei erfüllt werden.

1. Die Daten sollen möglichst nah am Benutzer verteilt werden d.h die Lokalität der Benutzer soll bei der Verteilung der Daten berücksichtigt werden, dadurch sollen die Paketlaufzeiten reduziert werden.
2. Der Service soll das Verteilungsproblem von IP-Multicast im Internet überbrücken. Es soll eine hohe Erreichbarkeit und Verfügbarkeit des Services gewährleistet werden.
3. Ein attraktives Business-Model für den Einsatz in der Cloud. Das Buissness-Model sollte für den Content-Provider und die Benutzer des Services transparent und unkompliziert sein.
4. Robuste Software Architektur die ohne Beschränkungen in der Cloud verteilt werden kann und selbst konfigurierend ist. Der Service soll auf Auslastungen reagieren und die Hardwareressourcen dynamisch vergrößern oder freigeben können.

Anforderungen an die Cloud

Wir nehmen an, dass die Cloud-Infrastruktur so gut verteilt ist, das Rechenzentren in der Nähe von den Benutzern verfügbar sind. Diese Anforderung ist wichtig damit die Paketlaufzeiten für den Dienst möglichst gering gehalten werden können. Des weiteren nehmen wir an, dass die Datenübertragung innerhalb der Cloud zuverlässig funktioniert und über eine hohe Bandbreite verfügt. Zudem sollte der Datentransfer kostengünstig innerhalb der Cloud sein. Nimmt man Amazon EC2 als Beispiel so ist der Datentransfer innerhalb der Rechenzentren und Region kostenlos, jedoch wird der Datentransfer unter den Regionen mit einem Preis von 0,01 Dollar pro GB berechnet [Amazon (2011)]. Einen beispielhaften Aufbau der Amazon Ec2 Infrastruktur ist in Abbildung 5 dargestellt.

Als Cloud-Infrastruktur haben wir uns für Amazon Ec2 entschieden. Amazon Ec2 ist ein IaaS Provider der freie Entscheidungen über die Auswahl des Images lässt und vollen Root zugriff gewährt. Andere Cloud Provider wie z.B Google App Engine und Microsoft Azure beschränken den Softwareentwickler zu stark, so ist Google App Engine z.B allein für den Gebrauch von Web Services ausgelegt. Ein weiterer Vorteil bei der Benutzung von Amazon Ec2 ist Eucalyptus [Eucalyptus (2011)] eine zu Amazon Ec2 kompatible Open-Source Cloud-Software. Mit Hilfe von Eucalyptus können wir die Software in einem kleineren Rahmen testen, ohne den Kosteneinsatz den längere Test mit Amazon Ec2 hervorrufen würden.

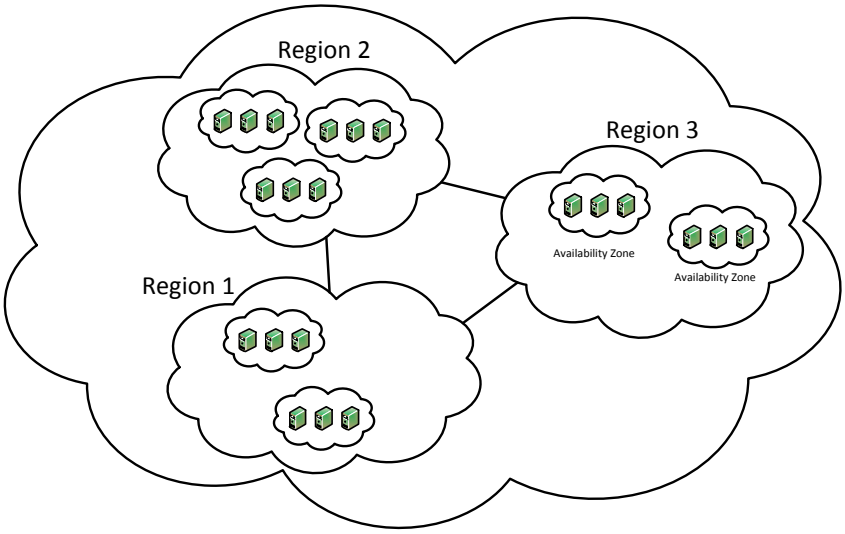

Abbildung 5: Amazon Ec2 Infrastruktur

4.2 Aufbau der Software

Die Software besteht aus drei Komponenten unterteilt einer API, einem Gateway und einem Multicast-Backbone.

- **Die API** Für den Gebrauch des Multicast-Dienstes benötigt der Softwareprogrammierer eine Programmierschnittstelle um eine Gruppenkommunikationsoftware zu entwickeln. Zu diesem Zweck wird ein Cloud-Computing Multicast-Module für die HAMcast-API entwickelt. Über die API hat der Softwareentwickler dann Zugriff auf den Cloud-Service und weitere alternative Multicast-Protokolle.

- **Das Gateway** Die Daten können auf zwei verschiedene Arten vom Benutzer abgerufen werden. Erstens, die Daten werden direkt über eine Unicast-Verbindung (Tunnel, http, Download) abgerufen. Zweitens, sollte mehr als einen Benutzer in einem lokalem Netzwerk exsistieren besteht die Möglichkeit ein Gateway einzurichten. Dieses Gateway dient dann als Schnittstelle zwischen dem Multicast-Backbone und dem lokalem Netzwerk. Das Gateway kann dazu verwendet werden die Daten über Multicast an eine Gruppe von Empfänger zu verteilen. Vorteil dieser Methode ist eine erhebliche Kostenersparnis für die Benutzer, da der Datentransfer aus der Cloud heraus und die damit verbunden Kosten auf diese Weise minimiert werden.

- **Multicast Backbone** Der Multicast-Backbone spannt einen Multicast-Verteilungsbaum innerhalb der Cloud auf. Der Backbone muss sich selbst konfigurieren d.h die Hardware Ressourcen abhängig von der Last eigenständig zu Skalieren und gegebenenfalls neue Instanzen in den Availability Zone und Regionen starten können. Die Daten sollten dabei immer nah an den Benutzern verteilt werden. Die Abbildung 6 zeigt anschaulich den Aufbau des Multicast-Backbones. Als Multicast-Verteilungsalgorithmus kann ein Application Layer Multicast wie z.B Scribe verwendet werden [Castro u. a. (2002)]. Um Kosten einzusparen sollte der Multicast-Baum so konfiguriert sein das der Datentransfer zwischen Regionen minimal ist. Damit diese Kriterium erfüllt werden kann muss eine Anpassung des Multicast-Algorithmus vorgenommen werden.

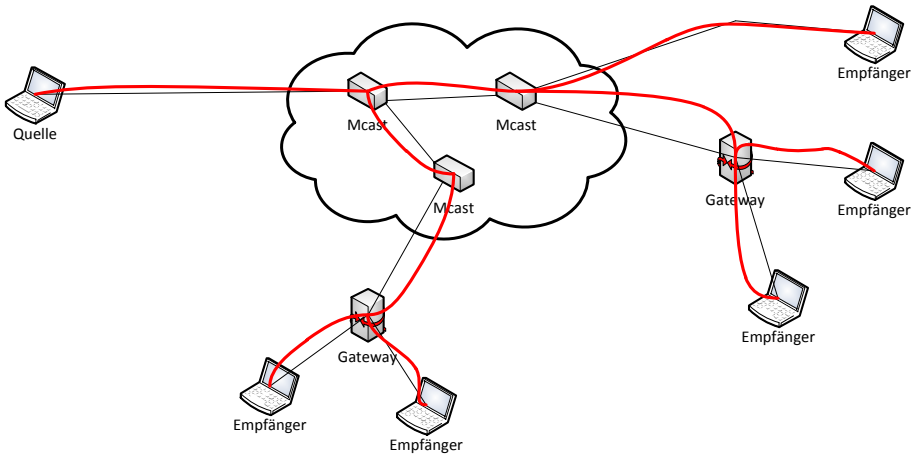

Abbildung 6: Multicast Backbone

4.3 Business Model

Als Business Model für den Multicast Backbone soll ein „Provider Driven Distribution System“ eingesetzt werden. Bei dem „Provider Driven Distribution System“ wird die Multicast- Infrastruktur komplett von einem Content-Provider verwaltet und eingerichtet. Die Abrechnung mit dem Cloud-Provider und den Benutzern des Dienstes erfolgt ausschließlich über den Content-Provider.

Provider Driven Distribution System

Um das Business Model zu veranschaulichen wird im folgenden der Ablauf in einem Scenario dargestellt. Ein Content-Provider möchte einen skalierbaren Daten Verteilungsdienst, einer unbekannte und flexible Anzahl an Kunden, zur Verfügung stellen. Der Content-Provider richtet dafür einen Initialen Multicast-Backbone in der Cloud ein, die Anzahl und die Verteilung der Instanzen innerhalb der Cloud werden von Provider übernommen. Bei der Initialen Verteilung sollten Faktoren wie die erwartete Anzahl an Benutzern und deren Standorte berücksichtigt werden. Der Backbone organisiert sich danach selbständig zu einem Multicast- Verteilungsbaum. Die Interpretation der Daten wird durch eine Software übernommen die der Content-Provider den Benutzern zur Verfügung stellen sollte. Die Software sollte als Schnittstelle zur Cloud dienen und kann einerseits direkt über eine Unicast-Verbindung (z.B einen Webservice) auf die Daten zugreifen oder ein Gateway instanziieren das die Daten dann lokal an eine Gruppe von Benutzern verteilen kann.

Vorteile

- Ein Upstream pro Content, nur der Content-Provider ist dazu berechtigt Daten an eine Gruppe zu senden.
- Kostengünstige Verteilung von Daten innerhalb der Cloud, das pay-per-use Prinzip der Cloud kann je nach Benutzung des Services wesentlich kostengünstiger als eine eigene IT-Infrastruktur sein.
- Datentransfer aus der Cloud heraus wird durch die Anzahl der Benutzer limitiert, wodurch das System für den Content-Provider transparent gestaltet ist.
- Einfacher Zugang zu den Daten mit Option auf lokalen Multicast
- Komplett Skalierbares Kostenmodell
- Einfache Kunden, Anbieter Geschäftsbeziehung

Nachteil

- Ein Service Anbieter Geschäftsmodell für die Nutzung des Dienstes wird benötigt, sowie die Möglichkeit zur Überwachung des Datentransfer, damit eine korrekten Abrechnung mit dem Kunden möglich ist.

4.4 Risiken

- Die Daten zwischen den einzelnen Multicast-Backbone Instanzen werden über Unicast-Verbindungen übertragen, ob so eine Erfüllung der „quality of service“ für Echtzeitdaten wie z.B für Videostreaming möglich ist werden erste Tests zeigen.
- Der Datentransfer zwischen den Regionen muss minimal gehalten werden um Kostengünstig arbeiten zu können. Damit sind sowohl die Daten die vom Content-Provider versendet werden gemeint als auch der Nachrichtenaustausch zwischen den einzelnen Instanzen innerhalb der Cloud, sowie die Daten die an den Kunden weitergeleitet werden. Dieses Kriterium ist eines der wichtigsten um die Kosten für den Dienst zu minimieren und entscheidet über die Akzeptanz bei Providern und deren Kunden.
- Dynamische Anpassung des Systems an die Auslastung die durch den Benutzer hervorgerufen wird muss gewährleistet werden. Eine Schlechte Skalierbarkeit des Dienstes würde den Nutzen schmälern.
- Überzeugendes Business Model für den Gebrauch des Multicast-Dienstes. Sollte das hier vorgeschlagene Business Model von den Content-Providern und deren Kunden nicht Akzeptiert werden könnte der Dienst und kein geeignetes Modell gefunden werden kann führt dies zum Scheitern des Dienstes.

5 Zusammenfassung

Ziel der Ausarbeitung war es ein Software-Konzept für einen virtualisierten, kostengünstigen und hoch skalierbaren Multicast-Dienst zu erstellen. Wenn der Cloud-Provider die erwarteten Anforderungen an die Cloud erfüllen kann, sollte es möglich sein das hier vorgestellt Software-Konzept umzusetzen. Was beim jetzigen Entwicklungsstand der Cloud-Services jedoch ein unvermeidbares Risiko bleibt, ist die Bindung der Infrastruktur an einen einzigen Cloud-Provider. Das hier vorgestellt Business Model sollte sowohl für den Content-Provider als auch für die Benutzer leicht überschaubar und umzusetzen sein. Ob das vorgeschlagene Business Model jedoch von den Benutzer und Conten-Provider akzeptiert wird ist zum jetzigen Zeitpunkt nicht vorherzusagen.

Literatur

[Amazon 2011] AMAZON: *Amazon EC2*. 2011. – URL http://aws.amazon.com/de/ec2/. – abgerufen 28.02.2010

[Armbrust u. a. 2010] ARMBRUST, Michael ; FOX, Armando ; GRIFFITH, Rean ; JOSEPH, Anthony D. ; KATZ, Randy H. ; KONWINSKI, Andy ; LEE, Gunho ; PATTERSON, David A. ; RABKIN, Ariel ; STOICA, Ion ; ZAHARIA, Matei: A view of cloud computing. In: *Commun. ACM* 53 (2010), Nr. 4, S. 50–58. – URL http://dblp.uni-trier.de/db/journals/cacm/cacm53.html#ArmbrustFGJKKLPRSZ10

[Castro u. a. 2002] CASTRO, Miguel ; DRUSCHEL, Peter ; KERMARREC, Anne-Marie ; ROWSTRON, Antony: SCRIBE: A large-scale and decentralized application-level multicast infrastructure. In: *IEEE Journal on Selected Areas in Communications* 20 (2002), Nr. 8, S. 100–110

[Diot u. a. 2000] DIOT, Christophe ; LEVINE, Brian N. ; LYLES, Bryan ; KASSEM, Hassan ; BALENSIEFEN, Doug: Deployment Issues for the IP Multicast Service and Architecture. In: *IEEE Network Magazine* 14 (2000), Nr. 1, S. 78–88

[Eucalyptus 2011] EUCALYPTUS: *Eucalyptus*. 2011. – URL http://www.eucalyptus.com/. – abgerufen 28.02.2010

[Fouquet u. a. 2009] FOUQUET, M. ; NIEDERMAYER, H. ; CARLE, G.: Cloud computing for the masses. In: *Proceedings of the 1st ACM workshop on User-provided networking: challenges and opportunities* ACM (Veranst.), 2009, S. 31–36

[Meiling u. a. 2010] MEILING, Sebastian ; CHAROUSSET, Dominik ; SCHMIDT, Thomas C. ; WÄHLISCH, Matthias: System-assisted Service Evolution for a Future Internet – The HAMcast Approach to Pervasive Multicast. In: *Proc. of IEEE GLOBECOM 2010, Workshop MCS 2010*. Piscataway, NJ, USA : IEEE Press, December 2010

[Wählisch und Schmidt 2007] WÄHLISCH, Matthias ; SCHMIDT, Thomas C.: In Between Underlay and Overlay: On Deployable, Efficient Group Communication Services. In: AL., Thomas C. S. et (Hrsg.): *Proceedings of the TERENA Networking Conference 2007*. Copenhagen : TERENA, May 21-24 2007. – URL http://tnc2007.terena.org/programme/presentations/show.php?pres_id=63

[Wählisch und Schmidt 2009] WÄHLISCH, Matthias ; SCHMIDT, Thomas C.: Multicast Routing in Structured Overlays and Hybrid Networks. In: SHEN, Xuemin (Hrsg.) ; YU, Heather (Hrsg.) ; BUFORD, John (Hrsg.) ; AKON, Mursalin (Hrsg.): *Handbook of Peer-to-Peer Networking*. Berlin Heidelberg : Springer Verlag, December 2009. – URL http://www.springer.com/computer/communications/book/978-0-387-09750-3

[Wählisch u. a. 2010] WÄHLISCH, Matthias ; SCHMIDT, Thomas C. ; VENAAS, Stig: A Common API for Transparent Hybrid Multicast / individual. URL http://tools.ietf.org/html/draft-waehlisch-sam-common-api, July 2010 (04). – IRTF Internet Draft – work in progress