Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Bachelorarbeit

Julian Torborg

Entwicklung und Evaluierung eines adaptiven hybriden Multicast auf Basis von Cloud Networking

Fakultät Technik und Informatik
Department Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Julian Torborg

Entwicklung und Evaluierung eines adaptiven hybriden Multicast auf Basis von Cloud Networking

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung

im Studiengang Technische Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Thomas C. Schmidt
Zweitgutachter: Prof. Dr. Hans Heinrich Heitmann

Abgegeben am 19. Juni 2012

Julian Torborg

Thema der Bachelorarbeit

Entwicklung und Evaluierung eines adaptiven hybriden Multicast auf Basis von Cloud Networking

Stichworte

Multicast, H∀Mcast , Cloud Computing, Amazon EC2

Kurzzusammenfassung

Eine effiziente Datenverteilung und Gruppenkommunikation kann mit IP-Multicast auf der Netzwerkebene erledigt werden. IP-Multicast hat aber ein Verfügbarkeitsproblem im Internet. Overlay-Multicast-Verfahren lösen dieses Problem, indem die Multicast-Kommunikation auf Anwendungsebene stattfindet. Hybride Multicast-Technologien vereinen den Einsatz von IP-Multicast und Overlay-Multicast, um einen universellen Multicast-Dienst zu ermöglichen. Cloud Computing Anbieter bieten dem Kunden einen großen Bestand an Rechen -und Netzwerkressourcen an. Dieser werden auf Anfrage sofort bereitgestellt und können an verschiedenen geographischen Orte plaziert werden. In dieser Arbeit wird ein Multicast-Netzwerk über das Internet mittels H∀Mcast und Amazon EC2 aufgebaut.

Title of the paper

Development and evaluation of an adaptive hybrid multicast based on cloud networking

Keywords

Multicast, H∀Mcast , Cloud Computing, Amazon EC2

Abstract

An efficient data distribution, and group communication can be done at the network layer using IP Multicast. IP Multicast has a deployment problem in the Internet. Overlay-Multicast solves this problem by moving the multicast communication to the application level. Hybrid-Multicast-technologies combines the use of IP multicast and overlay multicast to provide a universal multicast service. Cloud computing vendors offer customers a large pool of computing and network resources. These will be provided immediately on demand and can be placed in different geographical locations. In this work, a multicast network over the Internet is builded using HAMcast and Amazon EC2.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einführung

In der Computertechnik gibt es Anwendungen, welche auf Gruppenkommunikation basieren. Der Kernpunkt dieser Anwendungen ist es, dass die gleichen Daten an eine Gruppe von Empfängern im Netzwerk übertragen werden. Multicast-Verfahren übernehmen die Aufgaben der Gruppenverwaltung und die Datenverteilung bei der Gruppenkommunikation. Besonders interessant ist Multicast bei datenlastigen Gruppenanwendungen wie IPTV und Videokonferenzen. Das Problem des Multicast ist es, dass dieser im Internet nur geringfügig bis gar nicht verfügbar ist. Somit können beispielsweise die Teilnehmer einer Videokonferenz nicht einfach ihren Datenstrom ins Internet schicken und diesen dort an eine Gruppe verteilen lassen. Overlay-Multicast-Verfahren lösen dieses Verfügbarkeitsproblem, indem die Multicast-Kommunikation auf Anwendungsebene stattfindet. Dadurch entsteht aber ein erhöhter Kommunikationsaufwand und bei leistungschwachen Knoten oder hohen Verzögerunsgzeiten zwischen den Knoten ist die Kommunikation eingeschränkt. An diesen Punkt kann Cloud Computing ansetzen. Durch Cloud Computing werden dem Kunden Ressourcen sofort auf Anfrage bereitgestellt und dies verteilt auf mehrere Standorte in der Welt. Somit kann in und zwischen den Rechenzentren eines Anbieters ein leistungsstarkes Overlay-Multicast Netzwerk aufgebaut werden. Knoten können je nach Bandbreitebedarf hinzugeschaltet oder entfernt werden. Die Endknoten verbinden sich standortbezogen zu den Knoten in der Cloud. Letztendlich kann auf die Weise ein eigenes leistungsstarkes Multicast-Netzwerk im Internet aufgebaut werden.

Zielsetzung

Im Rahmen dieser Arbeit wird ein tiefer greifender Einblick in Multicast und Cloud Computing gegeben. Es wird sich im späteren Verlauf genauer auf die Middleware H∀Mcast für die Multicast-Kommunikation und Amazon EC2 für die Cloud Computing Umgebung konzentriert. Das Ziel ist es, diese in Zusammenspiel zubringen und einen adaptiven hybriden Multicast auf Basis von Cloud Networking zu entwickeln.

2 Multicast

In der Netzwerktechnik ist Multicast die Übertragung einer Nachricht von einen Sender an eine Gruppe von Empfängern. Im direkten Vergleich zum Unicast müssen die Daten nicht jedem Empfänger einzeln zugeschickt werden. Beim Multicast sendet die Quelle nur einen einzigen Datenstrom, das Netzwerk bestimmt mit intelligenten Mechanismen, wo die Daten angefordert wurden und leitet sie nur zu den Empfängern oder Netzwerken, welche Interesse an diesen mitgeteilt haben. Die Nachforschungen des Cisco's Visual Networking Index vom Juni 2011 [1] haben ergeben, dass der IP-Datenverkehr bis 2015 auf 966 Exabytes pro Jahr oder 80.5 Exabytes im Monat steigen wird. Der Internet Video Datenverkehr hat den peer-2-peer(P2P) Datenverkehr im Jahr 2010 eingeholt, gegen Ende des Jahres 2012 wird Internet Video 50 Prozent des gesamten Internet Datenverkehrs der Endverbraucher ausmachen. Das Übertragen dieser großen Menge von Inhalten kann nicht ökonomisch getan werden durch die Verwendung von Unicast oder sogar Content Delivery Networks(CDN). Unicast verursacht Skalierungs-Probleme, je mehr Verbraucher die Daten anfordern, desto öfters müssen diese Daten erneut übermittelt werden. CDNs werden hautsächlich dazu genutzt um Inhalte geograpisch an verschiedenen Standorten bereitzustellen und sind am besten geeignet für on-demand-Inhalte. Multicast hilft die Übertragung dieser großen Datenmengen zu handhaben, allerdings muss dafür jeder Netzwerkabschnitt, jeder Router und Firewall die Multicasttechniken unterstützen. Das größte Problem sind die *last-mile-networks* zum Endverbraucher, in denen meistens nur die Unicast-Kommunikation möglich ist. Um diese Abschnitte zu überbrücken müssten Tunneling Mechanismen verwendet werden, welche die Multicastpakete in Unicastpakete kapseln. Desweiteren können Overlay-Netzwerke aufgebaut werden, um Multicast-Daten in Form von peer-2-peer Techniken zu übertragen.

2.1 IP-Multicast

IP-Multicast [12] ist eine Implementierung des Multicast auf IP-Ebene. Ein Sender schickt das IP-Datenpaket an eine Gruppe, die Netzwerkinfrastruktur übernimmt die Verteilung der Pakete an jedes Mitglied dieser Gruppe. Dafür ist es notwendig, dass die Router im Netzwerk multicastfähig sind.
Eine Gruppe wird durch eine IP-Adresse gekennzeichnet, für diese IP-Adresse gibt es einen vordefinierten IP-Adressbereich. Für IPv4 ist das die Netzklasse D, dieser definiert sich von 224.0.0.0 bis 239.255.255.255, davon sind die Adressen 224.0.0.0 bis 224.255.255.255 unter anderen für Routingprotokolle belegt. Der Adressbereich für die IPv6 ist FF00::/8. Die Zieladresse des IP-Datenpakets welche per IP-Multicast an eine Gruppe gesendet werden soll, ist daher eine IP-Adresse aus dem erwähnten Bereich.
Die Empfänger müssen dem nächstgelegenden Multicast-Router mitteilen, dass sie Interesse an Multicast-Daten einer bestimmten Gruppe haben. Dies erfolgt über die entsprechenden Protokolle, Internet Group Management Protocol (IGMP) [10] bei IPv4 und Multicast Listener Discovery (MLD) [25] bei IPv6. Die Join-Nachricht dient zum Beitreten und die Leave-Nachricht zum Verlassen einer Gruppe. Der nächstgelegende Multicast-Router nimmt die

Nachrichten entgegen und verwaltet die Gruppen-Mitgliedschaften, er sendet General-Querys um die Hosts aufzufordern, dass diese ihre aktuellen Gruppen-Mitgliedschaften mitteilen. Die TTL bei dieser Nachrichten beträgt 1, somit muss der nächtsgelegende Router Multicast unterstützen, zudem dient die TTL beim Versand eines Multicast-Pakets wie oft es durch die Multicast-Router weitergeleitet wird.
Die Multicast-Kommunikation kann über mehrere Router stattfinden. Zur Umsetzung des Routings und den damit verbundenen Aufbau eines Verteilungsbaumes der Router werden Multicast-Routing-Protokolle eingesetzt. Das erste war das Distance-Vector-Multicast-Routing-Protocol (DVMRP) [29], mittlerweile hat sich das Protocol Independent Multicast - Sparse Mode (PIM-SM) [14] durchgesetzt.
Die Zustellung von Multicast-Daten kann auf zwei Arten erfolgen, Any Source-Multicast (ASM) und Source Specific Multicast (SSM) [8]. Beim ASM bekommen die Empfänger alle Pakete, welche an die Gruppenadresse gesendet werden zugestellt. Somit können mehrere Sender Pakete an eine Gruppe senden, auch wenn sie kein Mitglied dieser Gruppe sind. Beim SSM hingegen wird die Quelladresse mit einbezogen. Die Zustellung an die Gruppenmitglieder ist begrenzt auf einen spezifizierten Sender. Es wird sich an einen sogenannten Channel angemeldet, welcher sich aus dem Paar Quelladresse und Gruppenadresse zusammensetzt.
Das verwendete Transportprotokoll ist das User Datagram Protocol (UDP) [22]. Es hat die notwendige Eigenschaft, dass es verbindungslos ist und keine in Hinblick auf Multicast unnötigen Bestätigungen gesendet werden, wie es beim Transmission Control Protocol (TCP) [24] der Fall ist. Allerdings kann damit nicht garantiert werden, dass alle Pakete intakt beim Empfänger ankommen.
Somit ist IP-Multicast eine besonders effiziente Technik zur Datenübertragung in Netzwerken, welche die geforderten Techniken unterstützen. Da liegt allerdings das Problem des IP-Multicast, dieser ist meistens auf lokale Umgebungen (LANs) beschränkt, denn im Internet ist der Einsatz von multicastfähigen Routern gering, jedenfalls in den *last-mile-networks* zum Endverbraucher gibt es größtenteils keine Unterstützung des IP-Multicasts. Dies liegt zum einen daran, das sich die Service Provider einen Ertrag für ihre Investitionen erwarten und beim Multicast bisher die entsprechenden Geschäftsmodelle fehlen. Zum anderen ist der Aufwand zur Gruppenverwaltung in den Routern für alle Endverbraucher schwer zu stemmen.

2.2 Overlay-Multicast

Die begrenzte Verfügbarkeit von IP-Multicast hat zu dem alternativen Ansatz Overlay Multicast geführt. Die Abbildung 1 veranschaulicht nochmal, dass bei einem Unicast-only Netzwerk, die Multicast-Daten nur bis zum nächsten Router kommen, der Multicast-Join endet mit einem *Timeout*. Beim Overlay Multicast wird die Implementierung von der Netzwerkebene in die Anwendungsebene verschoben (siehe Abbildung 2). Es besteht nicht mehr die Abhängigkeit von der Verwendung multicastfähiger Router, denn es wird nun ein Overlay-Netzwerk mittels peer-2-peer-Technik zwischen den Systemen aufgebaut und verwendet.

Über dieses P2P-Overlay Netzwerk findet die Kommunikation mit Multicasttechniken statt. Implementierungen dieses Ansatzes sind u.a. Scribe [11], CAN-Multicast [23] und BIDIR-SAM [28].

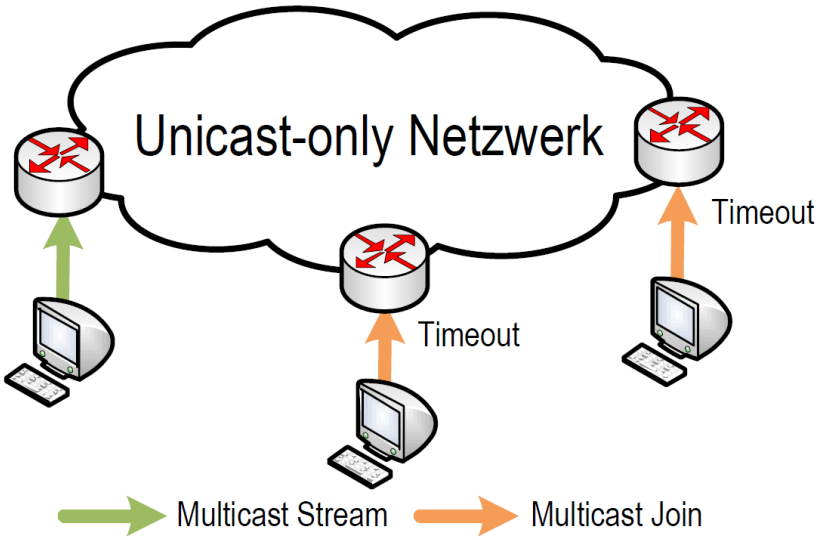

Abbildung 1: IP-Multicast in einem Unicast-Only Netzwerk

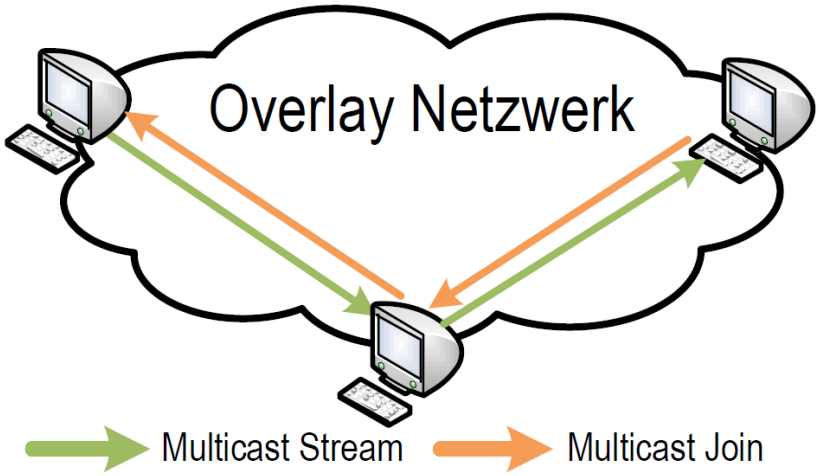

Abbildung 2: Overlay-Multicast

Scribe ist hochskalierend und geeignet für eine große Anzahl an Netzwerkknoten. Es basiert auf Pastry, ein skalierendes und selbst organisierendes peer-2-peer System für Suche und Routing. Scribe ist eine baumbasierte strukturierte Lösung, bei der alle Knoten dem demselben Overlay-Netzwerk beitreten und dann für jede Multicast-Gruppe ein eigener Verteilungsbaum aufgebaut wird. Jeder Knoten bekommt einen einmaligen Hashwert zugewiesen, ebenso die Multicast-Gruppen. Sollen nun Multicast-Daten versendet werden,

wird zunächst ein sogenannter Rendezvous-point bestimmt, dies ist derjenige Knoten dessen Hashwert am nähesten dem Hashwert der Multicast-Gruppe entspricht. Von diesen Rendezvous-points wird der Verteilungsbaum zu den Empfängern mittels reverse path forwarding aufgebaut. Die Empfänger senden eine Subscription-Nachricht an den Rendezvous-point unter Verwendung des Pastry Routing Algorithmus. Der Sender schickt die Daten an den Rendezvous-point und von dort aus werden diese dann über den Baum an die Empfänger verteilt.
Im Vergleich zum IP-Multicast ist der Einsatz beim Overlay-Multicast vereinfacht, da es sich auf fast jeder Netzwerkinfrastruktur einsetzen lässt, zudem lassen sich viele Overlay-Netzwerke parallel aufbauen. Allerdings ist die Effizienz geringer und der Overhead höher, denn zur Umsetzung des komplexen Routings auf Anwendungsebene erhöht sich der Rechenaufwand und der Netzwerkverkehr.

2.3 Multicast Tunneling

Multicast-Tunneling ermöglicht die Übertragung von Multicast-Pakete über Unicast-only Netzwerke. Im Folgendem werden zwei Verfahren vorgestellt.

2.3.1 Generic Routing Encapsulation

Generic Routing Encapsulation (GRE) [13] ist ein Tunneling Protokoll, welches alle Arten von IP-Paketen in IP-Pakete kapseln kann. Somit kann GRE auch genutzt werden um Multicast-Pakete über Netzwerke ohne Multicast Unterstützung zu transportieren. Es wird eine virtuelle Punkt-zu-Punkt Verbindung aufgebaut, welche IP Adressen aus dem privaten Adressbereich verwendet. Der Linux-Kernel bietet ein GRE Modul an. In Listing 1 sind die entsprechenden Shell-Befehle aufgeführt, um einen GRE-Tunnel anzulegen.

Listing 1: GRE-Tunnel System1

```
0 modprobe ip_gre
1 ip tunnel add tun0 mode gre remote 132.187.230.165 local 141.22.26.75 ttl
      225 dev eth0
2 ifconfig tun0 10.0.10.1/24
3 ifconfig tun0 pointopoint 10.0.10.2
4 ifconfig tun0 multicast
5 ifconfig tun0 up
```

Sofern das GRE Module nicht beim Systemstart geladen wird, muss dieses durch den Befehl *modprobe ip-gre* getan werden.
Der Befehl *ip tunnel* erstellt den Tunnel. In diesen Fall wurde der Name tun0 gewählt, der Modus ist gre, Angaben zur Remote IP-Adresse und der lokalen IP-Adresse, der TTL und dem physikalischen Gerät über das der Tunnel aufgebaut werden soll, werden gemacht. Im Anschluss werden über ifconfig die privaten IP-Adressen eingestellt, zudem die Aktivierung von Multicast. Am Ende wird das Interface aktiviert.

Für jedes Paket kommen beim GRE-Tunnel 20 Bytes für den neuen IP-Header und 4 Bytes für den GRE-Header hinzu.

2.3.2 Automatic Multicast Tunneling

Automatic Multicast Tunneling (AMT) [9] ist ein Protokoll zur Übermittlung von Multicast-Daten von Quellen aus Multicast-Netzwerken zu Empfängern, die keinen direkten Multicast Zugriff auf diese Netzwerke haben. Es wird eine UDP-basierte Kapselung und eine Unicast Replikation verwendet. Das Besondere ist, dass die Tunnel dynamisch eingerichtet werden. Im Vergleich zum GRE-Tunnel entfällt der manuelle Aufwand zur Einrichtung und Pflege des Tunnels zwischen zwei Standorten.
Die Abbildung 3 zeigt die generelle Architektur von AMT. Die wesentlichen Komponenten sind Gateway und Relay. Das Gateway befindet sich auf der Seite des Netzwerkes ohne Multicastzugang und das Relay auf der Seite des Multicast-Netzwerkes, getrennt sind diese durch ein Unicast-Netzwerk wie z.B. dem Internet. Es wird nach dem Client-Server Modell gearbeitet, das Gateway schickt eine Anfrage für einen Multicast-Stream an das Relay, dieses antwortet mit der Übermittelung des Multicast-Streams an das Gateway. AMT setzt weiterhin auf die Protokolle IGMP und MLD zur Verwaltung der Gruppen-Mitgliedschaften. Nach Erhalt des Membership Updates verwendet das Relay die Quell-IP-Adresse und den Quell-UDP-Port zur Übermittlung der Multicast-Pakete an das Gateway. Die UDP-basierte Kapselung übernimmt ein sogenanntes Pseudo-Interface auf beiden Seiten. Jedes Paket vergrößert sich um 20 Bytes für den IP-Header, 8 Bytes für den UDP-Header und 2 Bytes für den AMT-Header.
Somit ist es jedem System im Internet möglich einen dynamischen Tunnel zu einen Multicast-Netzwerk aufzubauen und Daten aus diesem zu empfangen. In [2] beschreibt Juniper Networks den Einsatz von AMT.

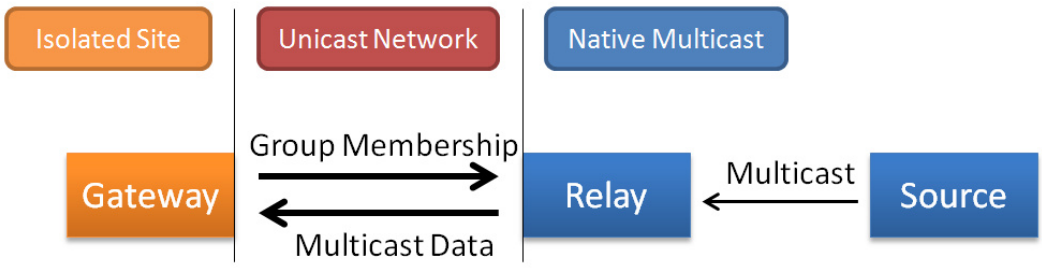

Abbildung 3: AMT Gateway und Relay

2.4 Hybrider Multicast

Hybride Multicast-Technologien vereinen den Einsatz von IP-Multicast und Overlay-Multicast. Somit können die Effizienzvorteile des IP-Multicast in multicastfähigen Netzwerken genutzt werden und die Verfügbarkeitsvorteile des Overlay-Multicast. Dabei werden IP-Multicast Inseln über ein P2P-Overlay zu einer Multicast-Domäne verbunden. Architekturen dieses Ansatzes sind Universal IP Multicast (UM) [30], Island Multicast (IM) [16], Scalable Hybrid Multicast (SHM) [17] und Hybrid Shared Tree (HST) [27].

2.4.1 H∀Mcast

H∀Mcast [18] stellt eine Systemarchitektur für einen universellen Multicast-Dienst bereit. Diese ist flexibel im Bezug auf die Übertragungstechnologien wie IPv4 und IPv6, sowie ASM und SSM. Im Vergleich zu anderen hybriden Ansätzen wird auf ein generisches Konzept gesetzt. Jede Gruppe existiert unter ihrem Namen verteilt auf Multicast-Domänen einer bestimmten Technologie. Ein Interdomain Multicast Gateway (IMG) vermittelt zwischen administrativen und technologischen Domänengrenzen (siehe Abb 4).

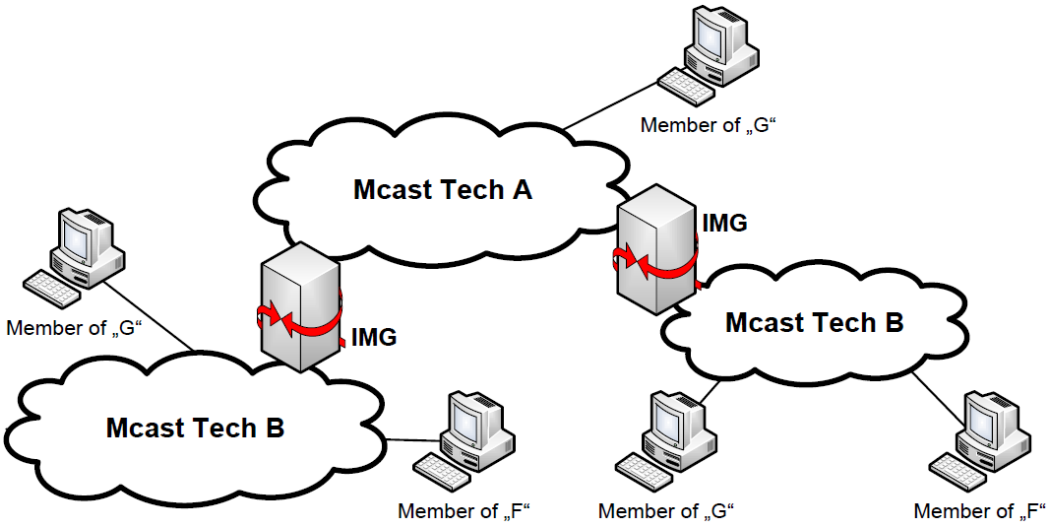

Abbildung 4: Netzwerk-Topologie mit mehreren Multicast-Domänen

H∀Mcast kombiniert eine namensorientierte Multicast-API mit einer Abstraktionsschicht in Form einer Middleware-Komponente. Die Middleware verwaltet Module für unterschiedliche Multicast-Technologien. Die derzeitig implementierten Module sind das *IP-Modul* und das *Scribe-Modul*. Die Middleware ist so gestaltet, dass neue Technologiemodule flexibel eingebaut werden können. Eine *Service-Discovery* hilft bei der Ermittlung verfügbarer Multicast-Technologien. Die gestartete Middleware stellt anschließend virtuelle Multicast-Interfaces bereit. Die Kommunikation mit der Middleware erfolgt über Inter Process Com-

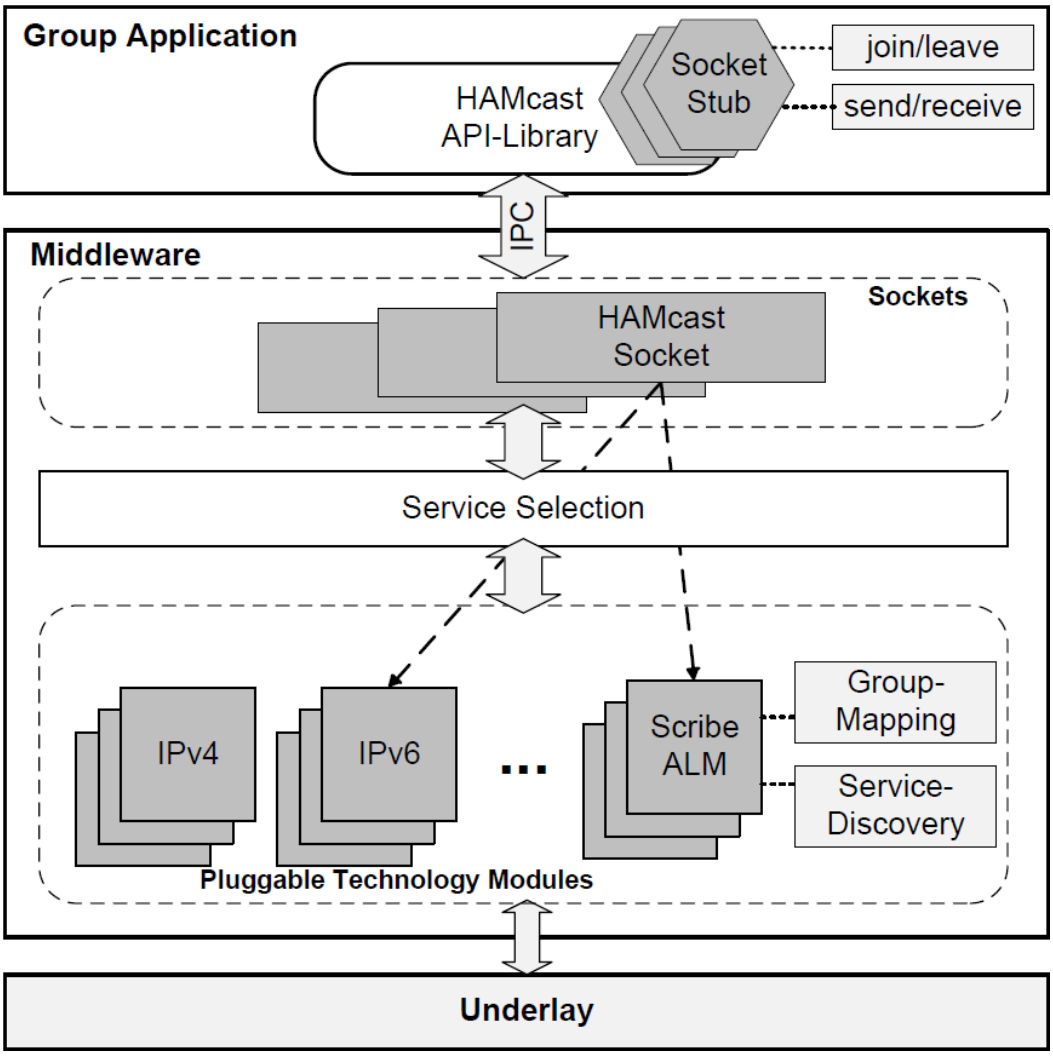

Abbildung 5: Aufbau und Komponenten von H∀Mcast

munication (IPC). Die IPC-Daten werden über den *Localhost*-Socket ausgetauscht. Eine abstrakte Programmierschnittstellte [26] dient zur Übergabe der Operationen von der H∀Mcast-basierten Anwendung. Dadurch werden einheitliche Aufrufe zur Gruppenkommunikationen unabängig von den eingesetzten Technolgiemodulen angeboten. Die Multicast-API ist derzeit als C++-Bibliothek sowie Java-Package für den Anwendungsentwickler verfügbar. Die Abbildung 5 zeigt den Aufbau und die Komponenten von H∀Mcast .
Mit der Konfigurationsdatei *middleware.ini* lassen sich die Module auswählen und einstellen (siehe Listing 2). Das IP-Modul dient zur nativen IP-Multicast Kommunikation, damit lassen sich die standardmäßigen Multicastpakete über die verfügbaren Interfaces verschicken. Dieses wird zur Kommunikation im lokalen Netzwerk (Underlay) mit Unterstützung von Multicast verwendet. Es stehen die IP-Version IPv4 und IPv6 zur Verfügung. Mit dem

Scribe-Modul lässt sich eine Overlay-Netzwerk basierend auf der Scribe-Technologie erstellen. Dieses wird zur Kommunikation zwischen Endpunkten verwendet, welche durch ein Netzwerk getrennt sind in dem keine Unterstützung von Multicast geboten wird. Es gibt einen Bootstrap-Knoten zu dem sich die Systeme erstmals verbinden um am Overlay-Netzwerk teilzunehmen.

Listing 2: middleware.ini

```
0  [global]
1  [ip_module]
2  file=../lib/libipmodule.so
3  ;Service discovery modes: off, light, full
4  discovery=off
5  ;Service discovery, ip versions: ipall, ip4, ip6
6  ipversion=ip4
7  [scribe_module]
8  file=../lib/libscribemodule.so
9  ;port=56765
10 ;bsnode=<bootstrap IP>
11 ;bsport=<bootstrap port>
12 ;maintenance=true
```

Eine Performanz-Evaluierung der H∀Mcast-Middleware in Bezug auf Paket- und Datendurchsatz kann in [19] eingesehen werden. Die Ergebnisse zeigen, dass die Perfomanz der Middleware mit dem IP-Stack mithalten kann und somit auch für datenlastige Gruppenanwendungen wie dem Videostreaming eingesetzt werden könnte.

3 Cloud Computing

Die folgende Beschreibung zum Cloud Computing hält sich überwiegend an die Definition des National Institute of Standards and Technology (NIST) [20] . NIST der US-Verwaltung hat eine Vorreiterrolle unter den Standardisierungsgremien, welches als erstes Gremium eine Standardisierungsroadmap für das Cloud Computing erarbeitet hat.
Cloud Computing ist ein Modell zur Bereitstellung eines allgegenwärtigen, bequemen und bedarfsorientierten Zugriffs auf einen gemeinsamen Pool konfigurierbarer IT-Resourcen wie Netzwerkinfrastruktur, Server, Speicher, Anwendungen und Dienste, die sich wiederum schnell mit minimalen Verwaltungsaufwand oder möglichst wenig Eingriffen durch die Service-Provider bereitstellen lassen. Außerdem gliedert sich das Modell in fünf grundlegende Merkmale, drei Service- und vier Bereitstellungs-Modelle.

Die fünf grundlegenden Merkmale sind On-demand self-service, Broad network access, Resource pooling, Rapid elasticity und Measured service.

On-demand self-service: Der Benutzer kann selbstständig auf die Ressourcen zugreifen, ohne dass ein händischer Eingriff durch den Provider benötigt wird.

Broad network access: Die Ressourcen sind über das Netzwerk erreichbar und der Zugriff erfolgt über Standardmechanismen von verschiedenen Thin- und Thick-Client-Plattformen.

Resource pooling: Die Ressourcen werden zusammengefasst um mehrere Anwender zu bedienen, diese werden dem Benutzer entsprechend der Nachfrage zugewiesen. Es gibt keine exakte Information zur Ansiedlung der Ressourcen, es kann aber gegebenenfalls auf höherer Ebene eine geographische Zone festgelegt werden.

Rapid elasticity: Die Ressourcen werden elastisch angeboten und können je nach Bedarf hoch- und runterskaliert werden. Für den Benutzer entsteht der Eindruck unbeschränkter Kapazität.

Measured service: Cloud Systeme bieten Messinstrumentarien an, um die Andwendung der Ressourcen zu kontrollieren und zu optimieren. Die Resourcennutzung kann somit beobachtet und berichtet werden.

Zu den drei Service-Modellen zählen Software as a Service (SaaS), Platform as a Service (PaaS), Infrastructure as a Service (IaaS).

Software as a Service (SaaS): Dem Benutzer werden Anwendungen angeboten, welche in einer Cloud Infrastruktur betrieben werden. Die darunterliegende Infrastruktur steht nicht zur Konfiguration zur Verfügung.

Platform as a Service (PaaS): Dem Benutzer wird es ermöglicht, seine eigene Anwendungen in einer Cloud Infrastruktur zu betrieben, gegebenenfalls samt Entwicklungsumgebungen, Bibliotheken, Diensten und Werkzeugen. Somit hat er die Kontrolle über seine eingesetzten Anwendungen, aber nicht über die darunterliegenden Ressourcen wie Betriebssysteme und Hardware.

Infrastructure as a Service (IaaS): Die Bereitstellung von Rechenkapazität, Massenspeicher, Netzwerkinfrastruktur und andere grundlegende IT-Ressourcen, auf denen es dem Benutzer ermöglicht wird, Software laufen zu lassen, zu denen Betriebssysteme und Anwendungen zählen. Der Benutzer verwaltet und kontrolliert nicht die darunterliegende Cloud Infrastruktur, (aber hat Kontrolle über Betriebssysteme, Speicher und eingesetzten Anwendungen).

Die vier Bereitstellungs-Modelle setzen sich aus Private Cloud, Community Cloud, Public Cloud und Hybrid Cloud zusammen.

Private Cloud: Die Cloud Infrastruktur ist bestimmt für den exklusiven Zugriff vorab definierter Nutzer einer Organisation. Management und Betrieb werden innerhalb eines Unternehmens oder einer gemeinsamen Organisation abgewickelt.

Community Cloud: Die Cloud Infrastruktur ist bestimmt für den Zugriff festgelegten Gruppen von Anwendern, deren Organisation gemeinsame Interessen vertreten.

Public Cloud: Die Cloud Infrastruktur ist bestimmt für den öffentlichen Zugriff. Die vielen verschieden Nutzer teilen sich die zugrunde liegende Infrastruktur. Der Eigentümer und Betreiber ist meist ein IT-Dienstleister.

Hybrid Cloud: Die Cloud Infrastruktur ist ein Zusammenschluss von zwei verschiedenen Cloud Infrastrukturen, die aber weiterhin zwei getrennte Einheiten bilden.

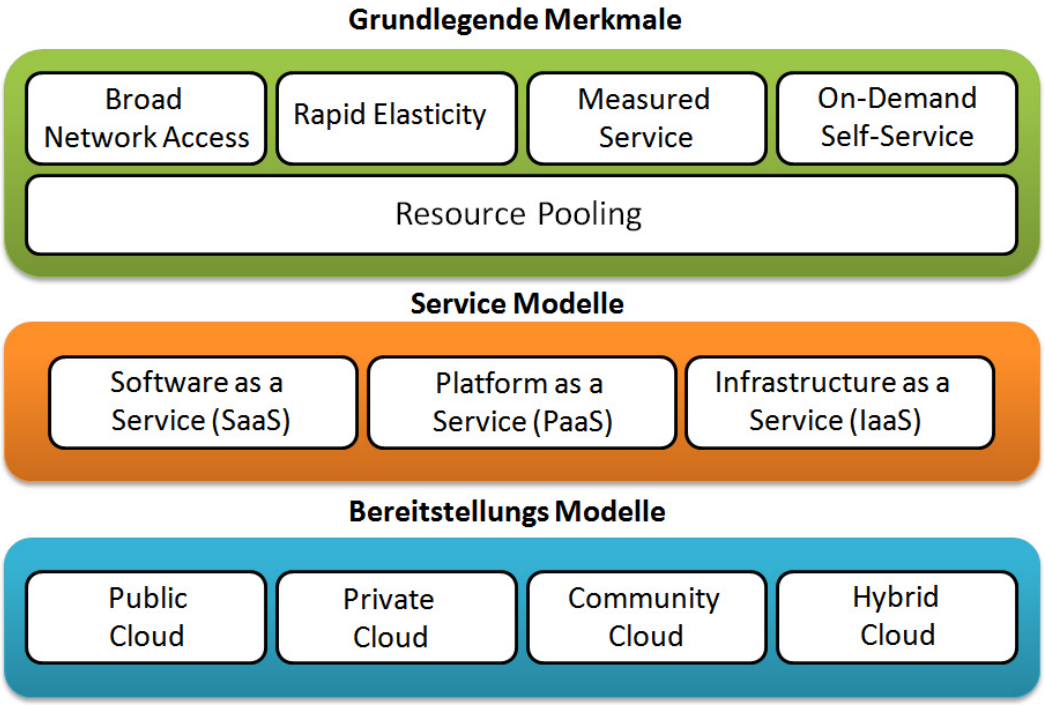

Abbildung 6: Cloud Modell nach NIST Definition of Cloud Computing

3.1 Amazon Web Services

Cloud Computing wird durch das Unternehmen *Amazon* mit Amazon Web Services (AWS) [3] angeboten, welcher eine Zusammenstellung mehrere Web Services ist. In diesen Zusammenhang wird sich speziell auf den Web Service Amazon Elastic Compute Cloud (Amazon EC2) konzentriert. Zunächst trotzdem noch ein Überblick zu Amazon Web Services.
Amazon hat viel Zeit und finanzielle Mittel investiert um eine großangelegte, effizente und zuverlässige IT-Infrastruktur aufzubauen für die riesige Online-Handelsplattformen *Amazon.com*. Mit AWS wird es dem Kunden ermöglicht diese Infrastruktur zu nutzen. Rechenleistung, Speicher und andere Dienste können entsprechend der Bedürfnisse in kürzester Zeit angefordert werden.
AWS wirbt mit den Eigenschaften Flexibilität, Kosteneffizenz, Skalierbarkeit, Sicherheit und Erfahrung. Die AWS-Infrastruktur basiert aus über 15 Jahren Erfahrung, welche durch den Aufbau von Amazon.com zusammen gekommen ist. AWS ist mit einer Reihe von Zertifizierungen im Bereich Sicherheit ausgestatt. Der Endanwender nutzt für seine Anwendung nur die Ressourcen der Cloud-Infrastruktur, welche er momentan benötigt. Es können nahzu alle Anwendungen betrieben werden. Die Programmiersprachen und Betriebssysteme können verwendet werden, welche bereits genutzt wurden oder am Besten für die Anwendungen des Endanweders geeignet sind. Somit entfällt die Investition in eine eigene IT-Infrastruktur und neue Technologien, stattdessen werden die genutzten Ressourcen verrechnet.

AWS setzt sich aus einer Reihe von Webservices zusammen, unten aufgeführt sind die Relevantesten.

- Amazon Elastic Compute Cloud (Amazon EC2)
- Amazon Simple Storage Service (Amazon S3)
- Amazon Elastic Block Store (EBS)
- Amazon Virtual Private Cloud (Amazon VPC)
- Amazon CloudFront
- Amazon Route 53
- Amazon Relational Database Service (Amazon RDS)
- Amazon SimpleDB

3.1.1 Amazon EC2

Der Webservice Amazon Elastic Compute Cloud (Amazon EC2) [5] bietet Rechenkapazität in einer Cloud Infrastruktur an, genauergesagt können virtuelle Server-Instanzen in den Rechenzentren von Amazon erstellt werden. Diese werden üblicherweise dazu genutzt, um ein eigenes Software-System zu hosten. Zur Nutzung gibt es eine Web-basierte Benutzeroberfläche, Kommandozeilen-Programme und eine Programmier-Schnittstelle. Server-Instanzen werden mit vorkonfigurierten Images gestartet. Ein Image setzt sich aus einen Betriebssystem und Software zusammen. Es wird eine große Auswahl verschiedener Images zur Verfügung gestellt. Zur Bestimmung der Ausstattung der Server-Instanzen gibt es Instanztypen, welche zum einen festlegen wie viel CPUs und Speicher verwendet werden. Zusätzlich können noch Sicherheitseinstellungen und Netzwerkzugang konfiguriert werden. Die Instanzen lassen sich innerhalb kürzester Zeit starten und beenden. Während des Betriebes kann die Auslastung beobachtet werden. Gezahlt wird dann für die tatsächlich genutzten Ressourcen. Das frei zugängliche Benutzerhandbuch von Amazon [4] liefert alle notwendigen Informationen zur Bedienung von EC2.

AWS Management Console (Web-basierte Benutzeroberfläche)

Die AWS Management Console (siehe Abbildung 7) ist eine Web-basierte Benutzeroberfläche zur komfortablen Verwendung der Amazon Web Services, sie veinfacht somit auch die Funktionsnutzung von Amazon EC2. Nach einem Login auf der AWS Management Console stehen unter dem Reiterpunkt EC2 die entsprechenden Funktionen zur Auswahl. Auf der Hauptseite gibt es die schnelle Möglichkeit zum Start einer Amazon EC2 Instanz, sowie die Einsicht zum aktuellen Status der Ressourcenverwendung der ausgewählten Region. Die

Region wird im Navigationsmenu gewählt, dort befinden sich die weiteren Punkte zur Verwendung und Einstellung der Funktionen für Instanzen, Images, Speicher, Netzwerk und Sicherheit. Somit wird ein einfacher Einstieg und eine komfortable Bedienung von Amazon EC2 ermöglicht, zur späteren Optimierung können Kommandozeilen-Programme und die Programmierschnittstelle verwendet werden.

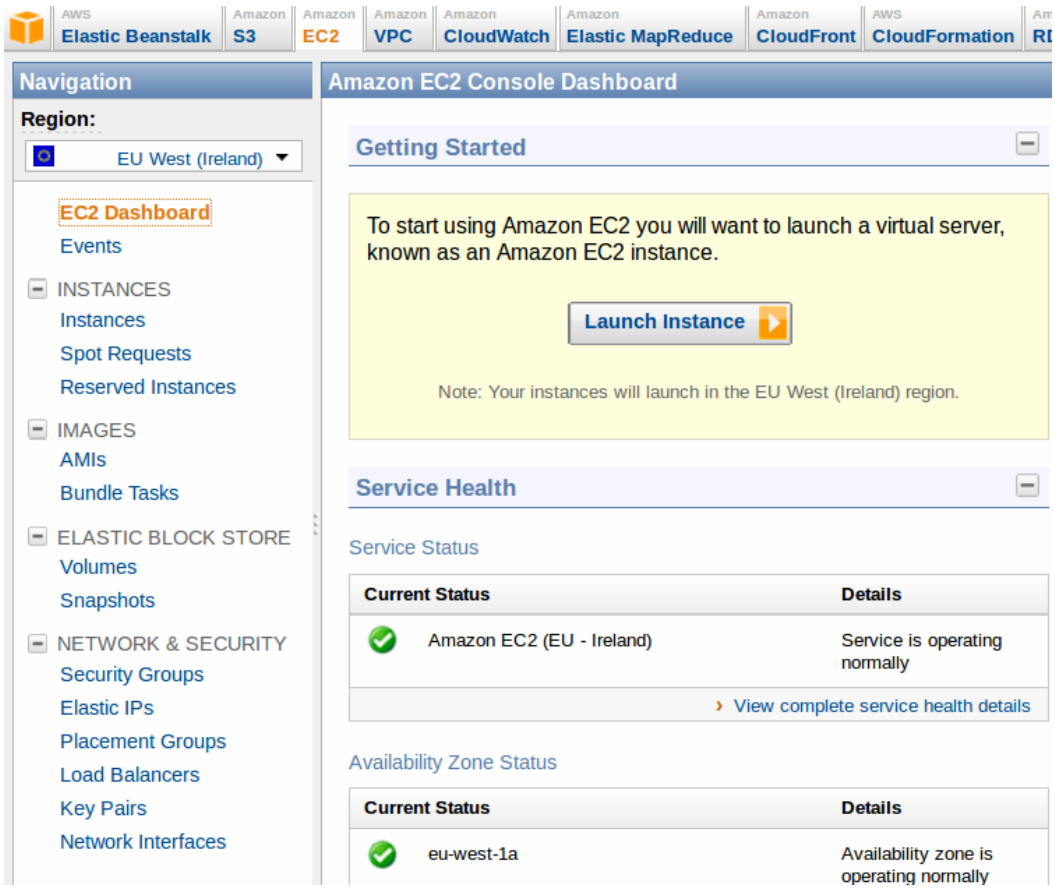

Abbildung 7: AWS Management Console

Amazon Machine Images

Ein Amazon Machine Images (AMI) ist ein verschlüsseltes Maschinen Image, welches alle Informationen enthält, um eine Instanz zu starten mit Betriebssystem und Software. Es gibt bereits eine große Auswahl verschiedener AMIs, Images von Amazon, Images von anderen EC2-benutzern oder selbsterstellte Images. Zu den angebotenen Betriebssystemen zählen unteranderem Amazon Linux AMI, Ubuntu Linux, SUSE Linux Enterprise, Red Hat Enterprise Linux und Windows Server. Bei der Software wird unteranderem folgendes angeboten, Apache HTTP, Hadoop und MySQL Enterprise. Zur Suche nach dem passenden Image und zur Verwaltung der Images hilft die AWS Management Console (siehe Abbildung 8).

Eigenschaft	EBS	Instance Store
Boot-Zeit	weniger als 1 Minute	weniger als 5 Minute
maximale Größe	1 TB	10 GB
AMI-Erstellung	ein Befehl	Aufwendig
Kosten	Instanzgebrauch, EBS-Volume, EBS-Snapshot	Instanzgebrauch, S3-Speicher

Tabelle 1: Speichertypen EBS und Instance Store

Ein wesentliches Merkmal eines Images ist die Speicherart, es gibt die beiden Kategorien Elastic Block Store (EBS) und Instance Store. Diese sorgt für Unterschiede in der Boot-Geschwindigkeit, AMI-Erstellung und Kosten. Die Tabelle 1 gibt einen Überblick.
Die Erstellung eines eigenen Images dient zur bestmöglichen Erfüllung der Anforderung vor dem Start der Instanz, indem die gewünschte Software installiert und Skripte zur Automatisierung von Prozessen eingestellt werden. Das Image wird in diesem Zustand gespeichert und danach können Instanzen von diesem gestartet werden. Die bereitgestellten Images können als Vorlage verwendet werden, zudem stellt Amazon EC2 die entsprechenden Funktionen bereit um möglichst schnell ein eigenes Image zu erstellen. Danach steht es einen frei, ob dieses nur privat oder öffentlich zur Verfügung steht.

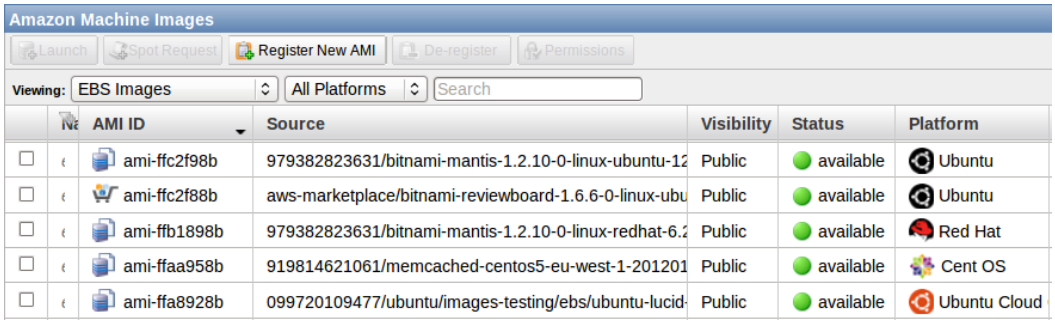

Abbildung 8: Verwaltung der Amazon Machine Images (AMI)

Instanz

Instanz wird das entstehende laufende System nach der Inbetriebnahme des AMIs genannt. Die Ressourcenausstattung der Instanz wird über die Auswahl des Instanztypes festgelegt. In Tabelle 2 sind die Eigenschaften von vier verschiedenen Instanztypen aufgelistet. Es wird die CPU-Ausstattung festgelegt, zu dessen einfachen Vergleich die Amazon EC2 Compute Unit (ECU) definiert wurde. Diese entspricht einem 1.7 GHz Xeon Prozessor. Es gibt den Arbeitsspeicher und den festen Speicher an und ob es sich um eine 32-Bit oder 64-Bit Platform handelt. Zusätzlich legt es die I/O-Performance fest. Ressourcen wie CPU und Speicher sind einer Instanz direkt zugeordnet, andere Ressourcen wie Netzwerk und

Type	Name	CPU	Speicher	I/O
Micro	t1.micro	bis 2 ECUs	613 MB	Gering
Small	m1.small	1 ECU	1.7 GB	Mittel
Medium	m1.medium	2 ECUs	3.75 GB	Mittel
Large	m1.large	4 ECUs	7.5 GB	Hoch

Tabelle 2: Instanztypen Micro, Small, Medium und Large

Speicherzugriff werden sich von mehreren virtuelle Maschinen auf einen System geteilt. Instanzen mit einer höheren I/O-Performance haben einen verbesserten Zugriff auf die gemeinsamen Ressourcen.

In der Abbildung 9 ist ein Ausschnitt aus der AWS Management Console zu sehen, welcher die laufende Instanzen auflistet. Jede Instanz bekommt eine eindeutige ID über welche sie später ansprechbar ist.

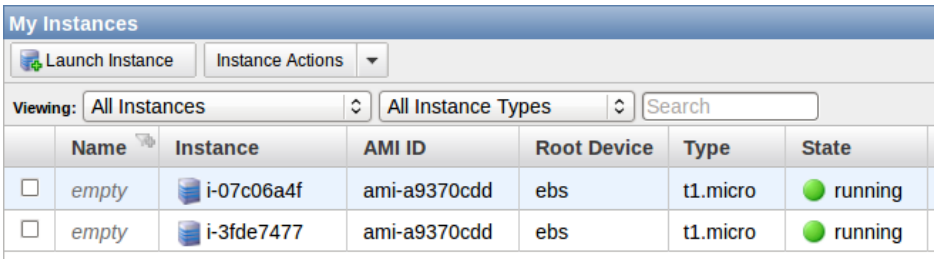

Abbildung 9: Auflistung der Instanzen

SSH Key Pair

Der Login auf linuxbasierten Instanzen erfolgt überlicherweise durch die Secure Shell (SSH). Dazu muss mindestens einmalig ein SSH Key Pair erstellt werden (siehe Abbildung 10). Vor dem Startvorgang der Instanz wird das gewünschte Keypair angegeben, beim Bootvorgang wird der Public-Key in die Instanz eingebaut, der Login ist nur Benutzern möglich, welche über den Private-Key verfügen.

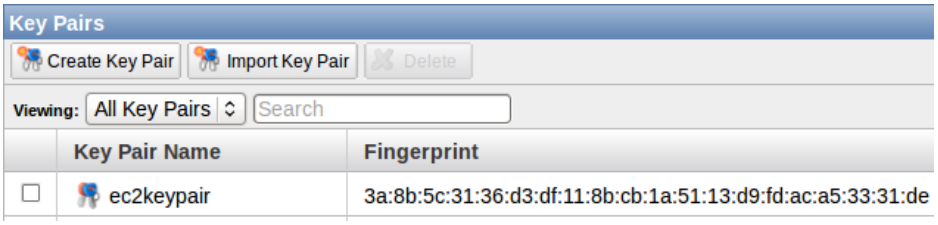

Abbildung 10: Keypairs in der AWS Management Konsole

Region	Name
Virginia	us-east-1
Nordkalifornien	us-west-1
Oregon	us-west-2
Sao Paolo	sa-east-1
Irland	eu-west-1
Singapur	ap-southeast-1
Tokio	ap-northeast-1

Tabelle 3: Regionen von Amazon EC2

Netzwerk

Einer Amazon EC2 Instanz werden immer zwei IP-Adressen zugewiesen, eine Private und eine Öffentliche, sie sind einander direkt zugeordnet über Network Address Translation (NAT). Die private IP-Adresse dient zur Kommunikation innerhalb des Amazon EC2 Netzwerkes. Dadurch wird sichergestellt, dass der Netzwerkverkehr optimiert übertragen wird in Bezug auf Bandbreite, Kosten und Verzögerungszeit. Die öffentliche IP-Adresse ist von Außen erreichbar und leitet den Netzwerkverkehr an die private IP-Adresse weiter und umgekehrt dient sie zur Kommunikation nach Außen. Es werden ausschließlich IPv4-Adressen verwendet. Im Amazon EC2 Netzwerk ist kein IP-Multicast verfügbar [7].

Sicherheitsgruppen

In Amazon EC2 gibt es das Konzept der Sicherheitsgruppen, sie handeln wie eine Firewall und kontrollieren den Netzwerkverkehr. Es werden Regeln für die Sicherheitsgruppen erstellt, welche den erlaubten eingehenden Verkehr bestimmen. Datenverkehr welcher nicht den Regeln entspricht wird blockiert. Bei der ausgehenden Datenübertragung gibt es keine Einschränkungen. Mit einer Sicherheitsgruppen-Regel kann genau bestimmt werden, welche Quelladressen über welches Protokoll und Port auf die Instanz zugreifen dürfen.

Regionen und Availibility Zones

Die Instanzen können an verschiedenen Positionen plaziert werden. Die Orte setzen sich aus Regionen und Availibility Zones zusammen. Regionen beschreiben geographische Orte in dem sich das Rechenzentrum befindet. Somit ist es möglich die Instanzen weltweit an verschiedenen Orten oder möglichst nahe an einen Kundenbereich zu platzieren. Availibility Zones sind eigene Bereiche innerhalb einer Region. In seltenen Fällen kann es vorkommen, dass in einer Availibilty Zone ein Fehler auftritt, werden die Instanzen in mehreren Availibilty Zones gestartet, kann somit der Ausfall eine Zone abgesichert werden.
Derzeit gibt es sieben Regionen weltweit, drei in Nordamerika, eine in Südamerika, eine in Europa und zwei in Asien (siehe Tabelle 3).

Typ	Irland	Virginia	Singapur
Micro	$0,020	$0,020	$0,020
Small	$0,085	$0,080	$0,085
Medium	$0,170	$0,160	$0,170
Large	$0,340	$0,320	$0,340

Tabelle 4: Preise für On-Demand Instanzen (Stand 15.06.2012)

Menge	Irland	Virginia	Singapur
Erstes 1 GB pro Monat	$0,000	$0,000	$0,000
Bis zu 10 TB pro Monat	$0,120	$0,120	$0,190
Nächste 40 TB pro Monat	$0,090	$0,090	$0,150
Nächste 100 TB pro Monat	$0,070	$0,070	$0,130

Tabelle 5: Preise für Internetdatenübertragung (Stand 15.06.2012)

Preisgestaltung

Die Preisgestaltung ist so ausgelegt, dass nur für die Ressourcen gezahlt wird, welche auch genutzt werden und es gibt keinen Mindesbetrag, welcher gezahlt werden muss. Die Abgaben lassen sich generell in drei Bereiche zerlegen, Instanzengebrauch, Datenübertragung und Speichernutzung. Die Preise variieren zwischen den verschiedenen Regionen.
Für den Erwerb von Instanzen gibt es drei Optionen, On-Demand Instances, Reserved Instances und Spot Instances. Bei On-Demand Instances wird für die Rechenkapazität pro Stunde abgerechnet und es wird keine langfristige Bindung gefordert. Somit können die Rechenkapazitäten flexibel je nach Anwendungsbedarf erhöht oder verringert werden und die Abrechnung erfolgt nur für den jeweiligen Stundensatz der genutzen Instanzen. Der Preis pro Stunde ist höher als der bei Reserved Instances. Eine Instanz wird für eine Dauer von einem oder drei Jahren reserviert, es wird eine Vorabgebühr gezahlt, der Preis pro Stunde ist dafür wesentlich geringer. Mit Spot Instances werden überschussige EC2-Ressourcen zu einen besonderen Preis angeboten.
Die eingehende Datenübertragung ist kostenfrei. Die ausgehende Datenübertragung wird pro Gigabyte berechnet. Der Preis pro Gigabyte senkte sich, sollte ein bestimmtes Datenvolumen überschritten sein. Der Datenverkehr innerhalb einer Availability Zone ist auch kostenfrei. Beim EBS-Datenträger wird im geringen Umfang der bereitgestellte Speicher pro Gigabyte im Monat berechnet, sowie für jede 1 Million I/O-Zugriffe.

Command Line Tools

Amazon stellt ein Kommandozeilen-Programm unter den Namen *Amazon EC2 API Tools* bereit. Es dient als Client-Interface zum Amazon EC2 Web Service mit dem über die Kommandozeile Befehle z.B. zum Starten einer Instanz übermittelt werden können. Die Auflis-

tung aller Befehle mit ausführlicher Beschreibung kann aus der *Amazon Elastic Compute Cloud Command Line Reference* entnommen werden [4].

API

Der Amazon EC2 Web Service ist auch über die Programmierschnittstelle erreichbar. Es werden Software Development Kits (SDK) für die Programmiersprachen Java, .Net und PHP bereitgestellt.
Für den direkten Zugriff auf die EC2-API können Query oder SOAP Anfragen verwendet werden. Query Anfragen bestehen aus HTTP-Nachrichten mit GET oder POST Parametern. Ein Web Services Description Language (WSDL) Dokument wird für die SOAP-Anfragen verwendet.

Metadaten

Aus einer Instanz können spezifische Metadaten der Instanz abgefragt werden, dazu zählen z.B. die Instanz-ID, private und öffentliche IP-Adressen oder der Hostname. Dies geschieht eine HTTP-Anfrage an die URI *http://169.254.169.253/latest/meta-data*. Der folgende Befehl würde die öffentliche IPv4 Adresse zurückgeben.

CURL http://169.254.169.254/latest/meta-data/public-ipv4

Zudem können einer Instanz vor dem Start sogenannte Userdaten mitgegeben werden, diese können dann aus der Instanz abgefragt werden um beispielsweise bestimmte Aktionen abhängig davon auszuführen. Der folgende Befehl gibt die Userdaten aus.

CURL http://169.254.169.254/latest/user-data

Monitoring

Der Dienst Amazon Cloud Watch stellt Messwerte zum CPU-Verbrauch und Netzwerkverkehr einer Instanz bereit. Im Basis-Modus werden die Werte alle fünf Minuten aktualisiert, gegen eine zusätzliche Gebühr findet im detailierten Modus, die Aktualiserung jede Minute statt.

4 Konzept

In den vorherigen beiden Kapiteln wurde ein Überblick zu Multicast und Cloud Computing gegeben. Der Kernpunkt beim Multicast ist die effiziente Datenverteilung an eine Gruppe von Empfängern. Beim Cloud Computing sind die wesentlichen Punkte, die sofortige Bereitstellung von Ressourcen aus einen riesigen Pool auf Anfrage, die daraus entstehenden Möglichkeiten zum Skalieren und die Nutzung verschiedener geographischer Orte verteilt auf der ganzen Welt. Das Ziel ist es, dass die beiden Gebiete in Zusammenhang gebracht werden, um einen verteilten, skalierbaren *Multicast-Backbone* in einer Cloud Computing Umgebung aufzubauen. Dadurch soll es zum einen möglich werden, die Vefügbarkeitsprobleme des IP-Multicast zu überbrücken. Systeme, welche ansonsten durch eine Unicastonly-Netzwerk getrennt werden, verbinden sich zur Cloud und würden darüber die Multicast-Kommunikation durchführen. Zum anderen könnten die Daten möglichst nahe am Endanwender bereitgestellt werden. Durch die Verteilung auf mehrere Standorte verbindet sich der Endanwender zum naheliegendsten. Die Verzögerungszeit würde sich verringern und die Daten müssten nicht unnötig über lange Strecken transportiert werden. Zudem lässt sich dieses Multicast-Netzwerk dynamisch in kürzester Zeit aufbauen und wieder abbauen. Die Leistungskraft ist durch die angbotenen Ressourcen nahezu beliebig hoch, lässt sich aber auf den tatsächlichen Bedarf anpassen. Die Abbildung 11 zeigt das grobe Konzept, indem sich die Endanwender zu Gateways in der Cloud verbinden für eine Multicast-Kommunikation.

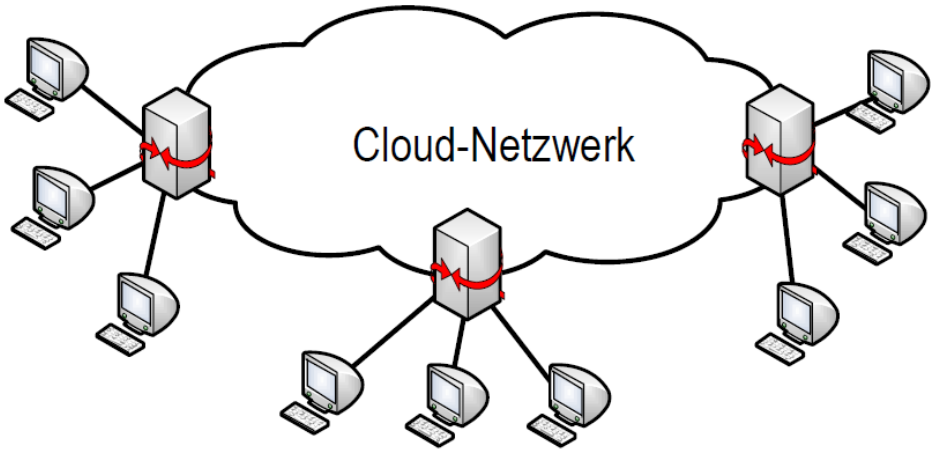

Abbildung 11: Cloud Netzwerk für Multicast

Einen Multicast-Dienst, welcher durch einen Cloud Computing Anbieter angeboten, gibt es bisher nicht. Der größte Cloud Computing Amazon bietet diesen auf jeden Fall nicht an und hat dies auch nicht in Aussicht gestellt. Daher liegt es an Externen ein System zu entwickeln um einen Multicast-Netzwerk auf Basis von Cloud Computing aufzubauen. Die Anwendungsansätze sind dabei vielseitig. Der Grundgedanke ist, dass ein System dadurch fähig ist, seine Daten an viele andere Systeme zu verteilen (siehe Abbildung 12).

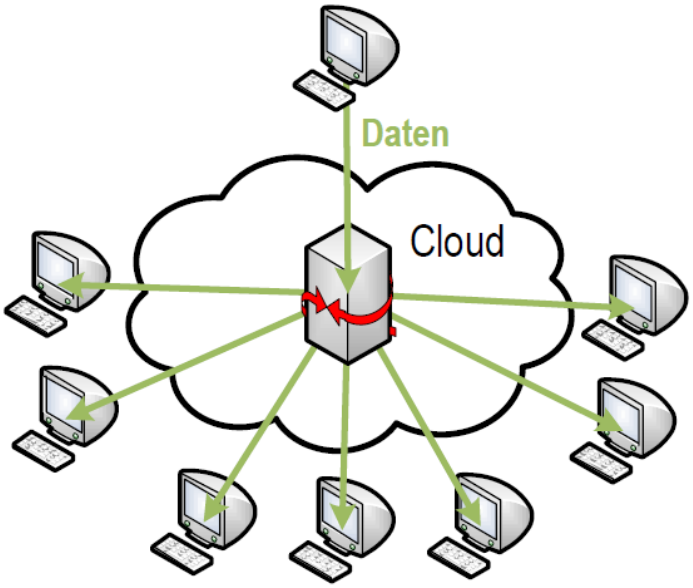

Abbildung 12: Datenverteilung durch die Cloud

4.1 Multicast-Netzwerk in der Cloud

Im folgenden wird das Konzept zum Aufbau eines Multicast-Netzwerkes mittels H∀Mcast und Amazon EC2 gezeigt. H∀Mcast stellt die Systemarchitektur für den Multicast-Dienst bereit. Amazon EC2 stellt die Infrastruktur in der Form von IaaS bereit. Die H∀Mcast-Software wird als Image bereitgestellt. Es werden mehrere Instanzen mit diesem Image in verschiedenen Regionen von Amazon EC2 gestartet. H∀Mcast regelt die Multicast-Kommunikation. Der IMG sorgt für die Vermittlung an den Außenknoten. Die Abbildung 13 zeigt den Aufbau.
Die Skalierung des Netzwerkes erfolgt durch das Hinzufügen und Entfernen von Instanzen in einer Region. Bei steigender Zahl von Endanwendern ist die Bandbreite nach einiger Zeit ausgereizt. Daher werden neue Instanzen gestartet, welche weitere Rechenleistung und Bandbreite liefern. Die Instanzen können wieder beendet werden sobald sich der Bandbreitenbedarf durch sinkende Anwenderzahl verringert. In den Regionen von Amazon EC2 wird kein IP-Multicast unterstützt, daher kommunizieren die Instanzen über ein Overlay-Multicast (siehe Abbildung 14).
Zu den Endnutzern außerhalb der Region besteht auch keine native IP-Multicast Anbindung. Diese sollen auch nicht direkt ins Overlay-Netzwerk einbezogen werden. Aus diesem Grund erfolgt die Multicast-Kommunikation über einen Multicast-Tunnel. Der IMG regelt dann wie erwähnt die Vermittlung zwischen Multicast-Netzwerk in der Cloud und dem Multicast-Tunnel (siehe Abbildung 15).

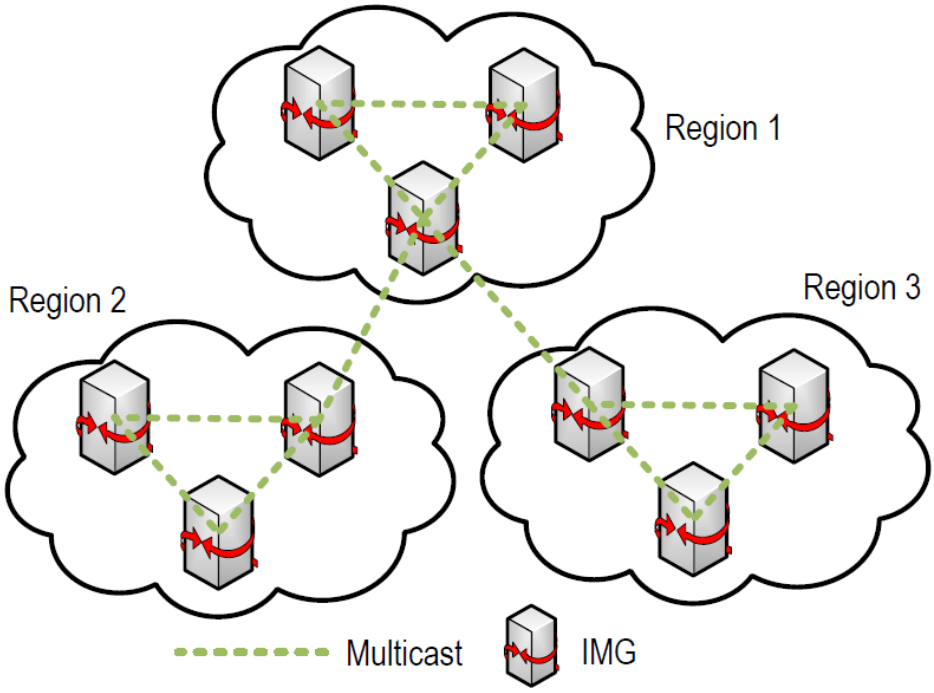

Abbildung 13: Multicast über mehrere Regionen

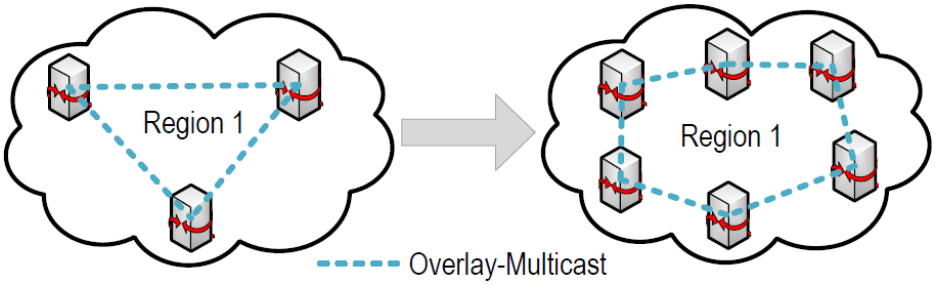

Abbildung 14: Skalierung durch Hinzufügen von Instanzen

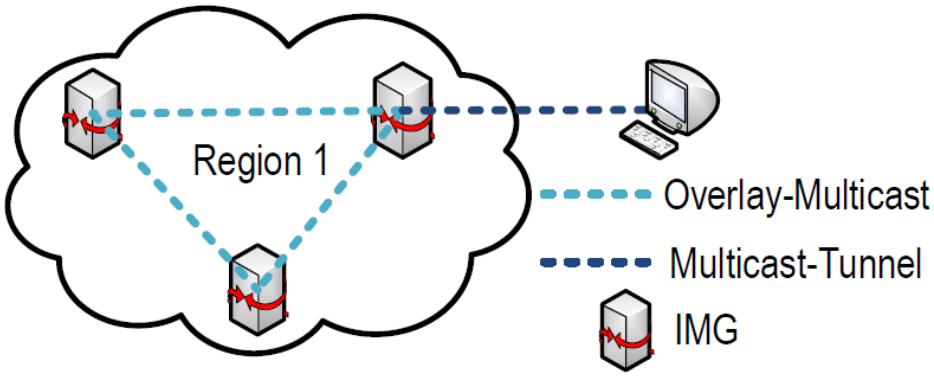

Abbildung 15: IMG zwischen Multicast-Tunnel und Overlay

5 Implementierung

In diesem Kapitel wird am Ende ein bestimmtes Testszenario für ein Multicast-Netzwerk in der Cloud erstellt. Zuvor werden Detaileinstellungen zur H∀Mcast-Middleware und Amazon EC2 behandelt, um die Vorraussetzungen dafür zu schaffen.

5.1 Einsatz der H∀Mcast-Middleware

Die in Abschnitt 2.4.1 vorgestellte H∀Mcast-Middleware soll auf den Instanzen in der Cloud eingesetzt werden. Diese ermöglicht den universellen Multicastdienst mit Techniken wie nativen IP-Multicast und den Overlay-Multicast. Das derzeitige Softwarepaket von H∀Mcast befindet sich in der Version *0.4.1*. Das Paket enthält die C/C++ Quellcode-Dateien. Die genaue Anleitung zur Durchführung des Build-Vorgangs kann von der H∀Mcast-Developer Webseite [15] entnommen werden. Nach Abschluss des Vorgangs sind die einzelnen Technologie-Module erstellt, sowie eine ausfürbare Datei zum Start der Middleware. Es werden das IP-Modul und das Scribe-Modul verwendet. Das IP-Modul dient zur nativen IP-Multicast Kommunikation über den Tunnel und das Scribe-Modul zur Overlay-Multicast Kommunikation zwischen den Instanzen.

5.1.1 Konfiguration der Middleware

Die Konfiguration der Middleware wird in der Datei *middleware.ini* erledigt. Hier werden die benötigten Module IP-Modul und Scribe-Modul aktiviert. Es wird IPv4 für das IP-Modul eingesetzt. Beim Scribe-Modul ist die Wahl des Bootstrapknoten wichtig. Hier steht die privaten IP-Adresse eines Knotens, welcher bereits Teil des Overlay-Netwerkes ist. Sollte dies der erste Knoten sein, bleibt der Eintrag auskommentiert. In Listing 3 ist die beispielhafte Konfiguration zu sehen, bei welcher die Instanz mit der privaten IP-Adresse *10.212.187.26* als Bootstrap-Knoten angegeben wird.

Listing 3: middleware.ini mit Bootstrap-Knoten

```
0  [ip_module]
1  file=../lib/libipmodule.so
2  ;Service discovery modes: off, light, full
3  discovery=off
4  ;Service discovery, ip versions: ipall, ip4, ip6
5  ipversion=ip4
6  [scribe_module]
7  file=../lib/libscribemodule.so
8  port=56765
9  bsnode=10.212.187.26
10 bsport=56765
11 ;maintenance=true
```

Scribe-Modul in Verwendung mit NAT

Diese Scribe-Implementierung ist nicht funktionsfähig in Verwendung mit dem NAT-Mechanismus. Amazon EC2 setzt allerdings den NAT-Mechanismus ein. Daher können nicht die öffentlichen IP-Adressen eingesetzt werden, sondern nur die privaten IP-Adressen und somit findet der Betrieb des Overlay-Netzwerkes einzig innerhalb einer Region statt. Die Verbindung der einzelnen Region muss somit in diesem Fall anders gelöst werden. Dazu werden feste Multicast-Tunnel zwischen zwei Instanzen in den verschiedenen Regionen eingerichtet. Hier gilt es zukünftig zu schauen, ob eine Implementierung für den Overlay-Multicast in Form eines H∀Mcast-Moduls entsteht, welche einsatzfähig mit NAT ist.

Anpassung der Hostdatei fürs Scribe-Modul

Das Scribe-Modul benötigt für den Einsatz einen Eintrag in */etc/hosts*, welcher die Adressierung in Bezug auf IP-Adresse und Hostname hergibt. Dieser Eintrag setzt sich bei einer EC2-Instanz aus der privaten IP-Adresse, den privaten DNS-Namen und den Hostnamen zusammen. Dieser würde wie folgt aussehen:

10.212.187.26 ip-10-212-187-26.eu-west-1.compute.internal ip-10-212-187-26

Die Erstellung dieses Eintrages kann durch ein Skript vereinfacht werden(siehe Listing 4). Die private IP-Adresse und der private DNS-Name werden mit den im Abschnitt 3.1.1 erwähnten Metadaten ermittelt. Nach dem Bootvorgang einer Instanz kann dieses Skript ausgeführt werden, um den entsprechenden Eintrag zu tätigen.

Listing 4: Skript zum Update von /etc/hosts

```
0 #!/bin/sh
1 PRIVIP=$(curl http://169.254.169.254/latest/meta-data/local-ipv4)
2 PRIVDNS=$(curl http://169.254.169.254/latest/meta-data/local-hostname)
3 HNAME=$(hostname)
4 echo $PRIVIP" "$PRIVDNS" "$HNAME >> /etc/hosts
```

Einsatz des IMGs

Der IMG vermittelt zwischen den verschiedenen Multicast-Domänen. Die Implementierung basiert auf der H∀Mcast-Middleware. In diesem Fall wird an den Schnittstellen zwischen Multicast-Tunnel und Overlay-Multicast eingesetzt. Dabei muss er einmal zwischen. Er wird benötigt, um die Daten zwischen dem entfernten System und der Cloud weiterzuleiten. Zudem regelt er die Weiterleitung zwischen den verschiedenen Regionen der Cloud.

5.2 Multicast-Tunneling

In 2.3 wurden bereits die Kernpunkte zum *Multicast Tunneling* beschrieben. Der Einsatz des Multicast-Tunneling kommt hier in zwei Fällen vor. Einmal zur Multicast-Kommunikation zwischen einem lokalen System eines Endanwenders und einer EC2-Instanz. Der zweite Fall ist zur Multicast-Kommunikation zwischen zwei EC2-Instanzen in verschiedenen Regionen. Als derzeitige Lösung bietet sich nur der GRE-Tunnel an. Für die Verbindung zwischen zwei festen EC2-Instanzen ist der GRE-Tunnel auch einsatzfähig. Bei der Verbindung zu den Endanwender wären aber leichtgewichtige und dynamische Lösungen gefragt, ähnlich dem AMT-Ansatz. Dieses könnte durch ein neues Technologie-Modul für H∀Mcast umgesetzt. Dadurch würde die derzeitige statische Konfiguration entfallen. Der Befehl *ip tunnel* dient unter Linux zur Erstellung eines GRE-Tunnels, die notwendigen Parameter sind zum einen die lokale IP-Adresse (local-address) und die IP-Adresse des anderen Systems (remote-address). Es gilt den NAT-Mechanismus für die Adressierung einer EC2-Instanz zu beachten, sodass einmal die private IP-Adresse und einmal die öffentliche IP-Adresse verwendet werden muss. In Listing 5 sind die jeweiligen Befehle für den Tunnel zwischen einem lokalen System und einer EC2-Instanz aufgeführt, unter Annahme das die IP-Adresse des lokalen Systems *141.22.26.75* ist und die EC2-Instanz hat die öffentliche IP-Adresse *46.51.144.100* und die private IP-Adresse *10.212.187.26*.

Listing 5: Befehle zum Tunnelaufbau zu einer EC-Instanz

```
0 Lokales System:
1 ip tunnel add tun0 mode gre remote 46.51.144.100 local 141.22.26.75
2 EC2-Instanz:
3 ip tunnel add tun0 mode gre remote 141.22.26.75 local 10.212.187.26
```

In Listing 6 sind die jeweiligen Befehle für den Tunnel zwischen zwei EC2-Instanzen aufgeführt. Die erste Instanz hat die öffentliche IP-Adresse *23.22.83.55* und die private IP-Adresse *10.4.115.51*. Die zweite Instanz hat die öffentliche IP-Adresse *79.125.45.169* und die private IP-Adresse *10.240.253.195*.

Listing 6: Befehle zum Tunnelaufbau zu einer EC-Instanz

```
0 EC2-Instanz 1:
1 ip tunnel add tun0 mode gre remote 23.22.83.55 local 10.240.253.195
2 EC2-Instanz 2:
3 ip tunnel add tun0 mode gre remote 79.125.45.169 local 10.4.115.51
```

Der GRE-Tunnel steht anschließend in Form eines Netzwerk-Interfaces bereit, z.B. mit dem Namen *tun0*. Dieses Interface kann von jeder Anwendung wie ein übliches Netzwerk-Interface genutzt werden, die H∀Mcast-Middleware greift auf dieses unter Verwendung des IP-Moduls zu.

5.3 Grundlegende Einstellungen für Amazon EC2

Regionen

Zur Verteilung der Multicast-Daten auf verschiedene geograpische Standorte werden drei Regionen von Amazon EC2 genutzt. Die Entscheidung fiel auf die Regionen Virginia (us-east-1), Irland (eu-west-1) und Singapur (ap-southeast-1). Die Endanwender verbinden sich standortbezogen zu der nächsten Region. Dementsprechend würden sich Anwender aus Amerika zu Virginia, aus Europa zu Irland und aus Asien zu Singapur verbinden. Das hat in diesem Fall beispielweise den Vorteil, dass eine Instanz aus Europa nicht jedem Anwender aus Amerika die Daten über die große Strecke zusenden muss. Stattdessen müssen sie nur einmal zur Region Virginia geschickt und von dort an die Endanwender aus Amerika verteilt. Die Konfigurationen der Regionen sind getrennt voneinander. Das bedeutet, dass Arbeiten zur Sicherheitsgruppe, Schlüsselpaar und Erstellung eines Images für jede Region einzeln gemacht werden.

Sicherheitsgruppe

Jeder Instanz muss eine Sicherheitsgruppe zugeordnet werden, um die erlaubten Netzwerkzugriffe auf die Instanz zu bestimmen. Hier müssen Netzwerkzugriffe für die Kommunikation im Overlay-Netzwerk und für den GRE-Tunnel erlaubt werden. Der Einfachheit halber werden Regeln angelegt, welche alle Zugriffe von Port 0 bis 65535 über TCP und UDP durchlassen.

Keypair

Ein Schlüsselpaar wird für jede der drei Regionen erzeugt, da dieses für den Start einer Instanz und dem späteren SSH-Login notwendig ist. Das Schlüsselpaar kann über die AWS Management Konsole oder mit den Amazon EC2 API Tools erstellt werden. Der private Schlüssel wird lokal gespeichert und bei einem späteren SSH-Login auf eine mit dem Schlüsselpaar gestartete Instanz mit angegeben.

ssh -i ec2keypair user@hostname

Instanztyp

Ein wesentlicher Parameter beim Start einer Instanz ist der Instanztyp zum Festlegen der Ressourcenausstattung. Es wird sich hier auf die Typen Micro, Small, Medium und Large konzentriert. Die bedeutenden Eigenschaften sind die Rechenleistung zum Betreiben der H∀Mcast-Middleware und die Netzwerkperformance zum Versenden der Daten. Die Instanz ist in diesem Fall umso leistungsfähiger, je mehr EC2 Recheneinheiten bereitstehen und je höher die I/0-Performance ist. Die Angaben zum festen Speicher sind nebensächlich, da dieser nicht benötigt wird.

5.4 Erstellen eines AMI mit H∀Mcast

Es wurde bereits im Abschnitt 3.1.1 erwähnt, dass eigene Images für Amazon-EC2 erstellt werden können. Dieses ist zur effizienten Nutzung meist unerlässlich, so auch in diesem Fall. Es wäre nicht sinnvoll, wenn nach jedem Start einer Instanz zuerst die umfangreiche Einrichtung des H∀Mcast-Softwarepakets getätigt werden müsste. Deshalb ist es das Ziel, ein auf dem EBS-Speicher basierendes Image zu erstellen, auf welchen H∀Mcast vollständig eingerichtet ist. Somit ist es einem danach möglich, schnell benutzerdefinierte Instanzen mit dem Image als Vorlage zu starten. Die Grundlage bildet ein Image aus dem EBS-Speicher mit dem Betriebssytem *Ubuntu 11.10*. Eine Instanz basierend auf dem genannten Image wird gestartet. Auf der gestarteten Instanz wird H∀Mcast installiert. Die Aufgaben zur Erstellung eines neuen AMI kann über die AWS Management Console oder den Amazon EC2 API Tools eingeleitet werden. In Abbildung 16 ist der entsprechende Aufruf *create image (EBS AMI)* im Kontextmenü der AWS Management Console gezeigt. Der jeweilige Kommandozeilebefehl der Amazon EC2 API Tools ist in Listing 7 aufgeführt.
Der anschließende Verarbeitungsprozess braucht etwas Zeit, da die Instanz heruntergefahren wird, das aktuelle EBS-Abbild wird kopiert und daraus wird das neue AMI erstellt und registriert. Dieses steht künftig unter einer einmaligen ID für die jeweilige Region zur Verfügung.

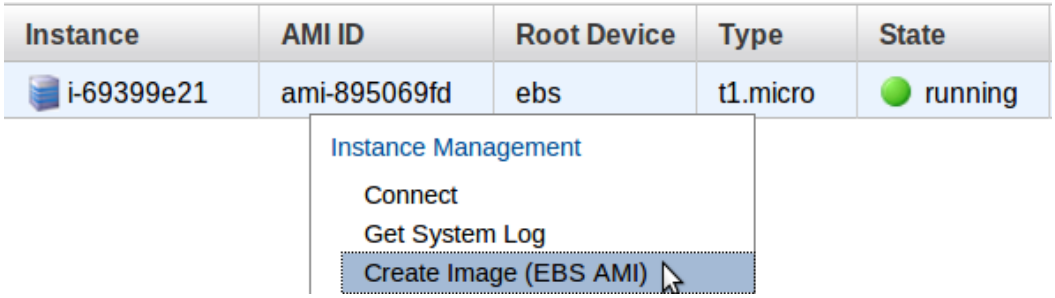

Abbildung 16: Erstellung eines Images mit der AWS Management Konsole

Listing 7: Erstellung eines Images mit der Amazon EC2 API Tools

```
0 ec2-create-image -n your_image_name instance_id
1 ec2-create-image -n MyAMI i-eb977f82
```

Listing 8: Konfigurationsskript für die Amazon EC2 API Tools

```
0 export JAVA_HOME=/usr/lib/jvm/java-6-openjdk/
1 export EC2_HOME=~/ec2-api-tools
2 export PATH=\$PATH:\$EC2_HOME/bin
3 export EC2_PRIVATE_KEY=~/.ec2/pk-O5ZE7FX5QY6IG7GJGZZFMCWU55EX4PFX.pem
4 export EC2_CERT=~/.ec2/cert-O5ZE7FX5QY6IG7GJGZZFMCWU55EX4PFX.pem
5 export EC2_URL=https://ec2.eu-west-1.amazonaws.com
6 #export EC2_URL=https://ec2.us-east-1.amazonaws.com
7 #export EC2_URL=https://ec2.ap-southeast-1.amazonaws.com
```

5.5 Software zum komfortablen Start einer EC2-Instanz

In diesen Abschnitt wird die Bedienung von Amazon EC2 ohne die AWS Management Console gezeigt, stattdessen wird ein Kommandozeilenprogramm und die Programmierschnittstelle verwendet.

5.5.1 Anwendung der Amazon EC2 API Tools

Die Amazon EC2 API Tools bieten wie in Abschnitt 3.1.1 angedeutet, die schnelle und komfortable Bedienung von Amazon EC2 über die Kommandozeile. Die Software basiert auf der Programmiersprache Java und für die einzelnen Funktionen enthält es Shellskripte für Windows und Linux. Im folgenden wird sich auf die Einrichtung unter Linux konzentriert. Das Softwarepaket wird als ZIP-File zum Download angeboten, danach folgt der Entpackvorgang, ein Installationsvorgang ist nicht notwendig. Es ist erforderlich, dass die entsprechenden Umgebungsvariablen gesetzt werden. Dazu gehört die Umgebungsvariable *JAVA_HOME* zum lokalisieren der Java Laufzeitumgebung. Die Variable *EC2_HOME*, um dem System den Pfad zu dem Programm mitzuteilen. Zur Authentifizierung verwendet das Programm einen Private-Key und ein X.509-Zertifikat, diese werden vom persönlichen AWS-Konto heruntergeladen. Die entsprechenden Umgebungsvariablen lauten *EC2_PRIVATE_KEY* und *EC2_CERT*. Zusätzlich kann noch die Endpoint-URL gesetzt werden zur Festlegung der zu verwendenen Region im Voraus, dies geschiet über *EC2_URL*. Es sei nun angenommen, dass die Amazon EC2 API Tools im Homeverzeichnis in den Ordner *ec2-api-tools* entpackt wurden und der Private-Key und das Zertifikat im Ordner *.ec2* des Homeverzeichnisses gespeichert wurden. Bei den Endpoints wird zwischen eu-west-1, us-east-1 und ap-southeast-1. Ein entsprechendes Konfigurationsskript zur Einrichtung der Umgebunsgvariablen ist in Listing 8 aufgeführt.
Nach der Einrichtung der Amazon EC2 API Tools ist es nun beispielsweise möglich, Befehle zum Starten oder Stoppen von Instanzen über die Kommandozeile auszuführen. In Zusammenhang dieser Arbeit ist es hilfreich einen Befehl zusammenzustellen, mit welchem es möglich ist eine Instanz mit dem selbesterstellten Image und der entspreched Sicher-

Listing 9: Kommmandozeilenbefehl zum benutzerdefinierten Instanzstart

```
0 ec2-run-instances ami-a9370cdd -t t1.micro -n 1 -k ec2keyirl -g sg-6
    a70801d
```

heitsgruppe, sowie Schlüsselpaar zu starten. In Listing 9 ist der Kommmandozeilenbefehl mit den jeweiligen Parametern aufgeführt, der Instanztyp ist in diesem Fall micro.

5.5.2 Anwendung des AWS SDK für Java

Die Amazon EC2 API Tools liefern alle grundlegenden Funktionen zur Kontrolle von Amazon EC2, allerdings sind diese nicht ausreichend, sofern der Anwender mehr ins Detail gehen und eigene Logik mit einbringen möchte. Dieses wird auf programmiertechnische Weise gelöst. Unterstützung bringen dabei die von AWS angebotenen SDKs.
Das Ziel ist es ein Programm zu entwickeln, welches eine Instanz mit dem H∀Mcast- Image in der gewünschten Region startet, auf den Abschluss des Bootvorgangs wartet und danach die SSH-Zugriffsdaten ausgibt. Die eingesetzte Programmiersprache ist Java, dementsprechend wird das AWS SDK für Java benutzt [6]. In diesem enthalten ist die AWS-Java-Bibliothek. Diese Bibliothek hilft enorm bei der Arbeit mit der Webservice-Schnittstellt von AWS, indem sie die Generierung der HTTP-Anfragen, sowie die Verarbeitung der XML-Rückgabewerte übernimmt und dieses dem Entwickler über einfache Funktionen zur Verfügung stellt. Die in dieser Arbeit eingesetzte Version ist *1.3.8*.
Die Authentifizierung mit der AWS-Java-Bibliothek erfolgt unter Verwendung der *Access Key ID* und des *Secret Access Key*, welche online über das AWS-Konto bezogen werden können.
Eine Konfigurationsdatei soll alle relevanten Einstellungen enthalten, sodass dort später Änderungen wie zum Beispiel zur *Access Key ID* oder zum *Instanztypen* gemacht werden können. In Listing 10 ist der Inhalt der Konfigurationsdatei zu sehen, welche in den Punkten *accesskey*, *secretkey* und *keyname* mit den persönlichen Daten gefüllt werden muss. Der entsprechende Programmierbefehl zum Auslesen der Datei ist in Listing 11 aufgeführt. Letztendlich gibt es für jede der drei Regionen eine Konfigurationsdatei. Beim Start des Programms wird die gewünschte Region als Parameter übergeben und danach können die Werte aus der entsprechenden Datei ausgelesen werden.

Listing 11: Laden der Konfigurationsdatei

```
0 Properties prop = new Properties();
1 prop.load(new FileInputStream("awsconfig.properties"));
2 String accesskey = prop.getProperty("accesskey");
3 String secretkey = prop.getProperty("secretkey");
```

Listing 10: Konfigurationsdatei für das Programm

```
#Insert your AWS Config
accesskey=<your access-key-id>
secretkey=<your secret-access-key>
endpoint=https://ec2.eu-west-1.amazonaws.com
zone=eu-west-1c
instancetype=t1.micro
imageid=ami-a9370cdd
secgroupid=sg-6a70801d
keyname=<your keypair-name>
user=ubuntu
```

Listing 12: AmazonEC2Client mit Zugangsdaten

```
AmazonEC2Client ec2 = new AmazonEC2Client(
                      new BasicAWSCredentials(accesskey, secretkey));
```

Die Klasse *AmazonEC2Client* stellt alle notwendigen Funktionen zur Bedienung von EC2 bereit. Dem Konstruktor müssen die Zugangsdaten mitgegeben werden(siehe Listing 12). Die Methode *runInstances* sorgt dafür, dass eine neue Instanz in Amazon EC2 erstellt und gestartet wird. Ein *RunInstancesRequest* enthält alle erforderlichen Informationen wie den Instanztypen, der Image-ID, der Zone, der Sicherheitsgruppe und dem Schlüsselpaar. Dieser wird der Methode als Argument übergeben, nach Abschluss des Methodenaufrufs und somit auch dem Start der neuen Instanz wird ein *RunInstancesResult* zurückgegeben. Dieser verfügt über alle Daten der gestarteten Instanz, wie beispielsweise der zugewiesenden Instanz-ID. In Listing 13 ist die genaue Verwendung gezeigt. Die neue Instanz wurde nun gestartet, der Abschluss des Bootvorgangs braucht allerdings noch eine unbekannte Zeitspanne. Die Abfrage der aktuellen Informationen der Instanz ist über die Methode *describeInstances* möglich, dazu zählt auch der derzeitige Status wie *pending* oder *running*. Mit einer Schleife erfolgt die Abfrage des Status, bis dieser den Wert *running* aufweist(siehe Listing 14. Sobald sich die Instanz im Status *running* befindet, verfügt diese auch über eine öffentliche IP-Adresse.

Listing 14: Schleife zur Statusabfrage

```
do{
   TimeUnit.SECONDS.sleep(10);
   diresult = ec2.describeInstances(direquest);
   instance = diresult.getReservations().get(0).getInstances().get(0);
}while(!instance.getState().getName().equals("running"));
```

Listing 13: Start einer Instanz

```
0 RunInstancesRequest rirequest = new RunInstancesRequest()
1   .withInstanceType(instancetype)
2   .withImageId(imageid)
3   .withMinCount(1)
4   .withMaxCount(1)
5   .withPlacement(new Placement(zone))
6   .withSecurityGroupIds(secgroupid)
7   .withKeyName(keyname);
8 RunInstancesResult riresult = ec2.runInstances(rirequest);
```

Es wurden die wesentlichen Mittel beschrieben, um die gewünschte Funktionalität auf programmiertechnische Weise umzusetzen. Das Projekt wird als ausführbare JAR-Datei exportiert, diese kann von der Kommandozeile direkt ausgeführt werden(siehe Listing 15). Im gleichen Ordner befindet sich die Konfigurationsdatei, welche zuvor mit den persönlichen Daten ergänzt werden muss.

Listing 15: Start der JAR-Datei

```
0 > java -jar awsHamcaststarter.jar irland
1 Instance with ID i-bf5285f7 has been started.
2 ---
3 Instance i-bf5285f7(46.51.144.100) is running!
4 ssh -i ec2keyirl.pem ubuntu@46.51.144.100
```

5.5.3 Programm für den Testaufbau

Für den späteren Testaufbau wurde ein Programm erstellt, welches jeweils drei Instanzen in den Regionen Irland, Virginia und Singapur startet. Die grundlegende Basis bietet das im vorherigen Abschnitt entwickelte Programm. Der Ablauf ändert sich dahin, dass nacheinander in jeder Region drei Instanzen gestartet werden. Beim *RunInstancesRequest* ist daher der Parameter für *withMinCount* und *withMaxCount* der Zahl 3. Nach Abschluss der Startvorgänge aller Instanzen werden die Zugangsdaten aufgelistet. Somit ist es mit diesem Programm später möglich ohne großen Aufwand die benötigten Instanzen für den Testaufbau zu starten.

5.6 Testaufbau des Multicast-Netzwerkes

In den vorherigen Punkten zur Implementierung wurden die Vorraussetzungen für einen Testaufbau eines Multicast-Netzwerkes mit H∀Mcast in Amazon EC2 geschaffen. Dieser wird genutzt, um die Funktionalität zu demonstrieren, zu testen und zu analysieren. Mit dem in 5.5.3 vorgestellten Programm werden jeweils drei Instanzen in den drei Regionen gestartet. Die Instanzen werden innerhalb einer Region über Overlay-Multicast mittles Scribe zusammengeschlossen. Es wird ein GRE-Tunnel zwischen einer Instanz in Irland und einer in Virgina eingerichtet und einer zwischen zwei Instanzen aus Irland und Singapur. Die IMGs werden an den Schnittstellen entsprechend zur Weiterleitung der Daten eingerichtet. In Abbildung 17 ist der genaue Testaufbau gezeigt, beispielhaft mit einem Sender bei der Region Irland und zwei potenziellen Empfänger bei den Regionen Virginia und Singapur.

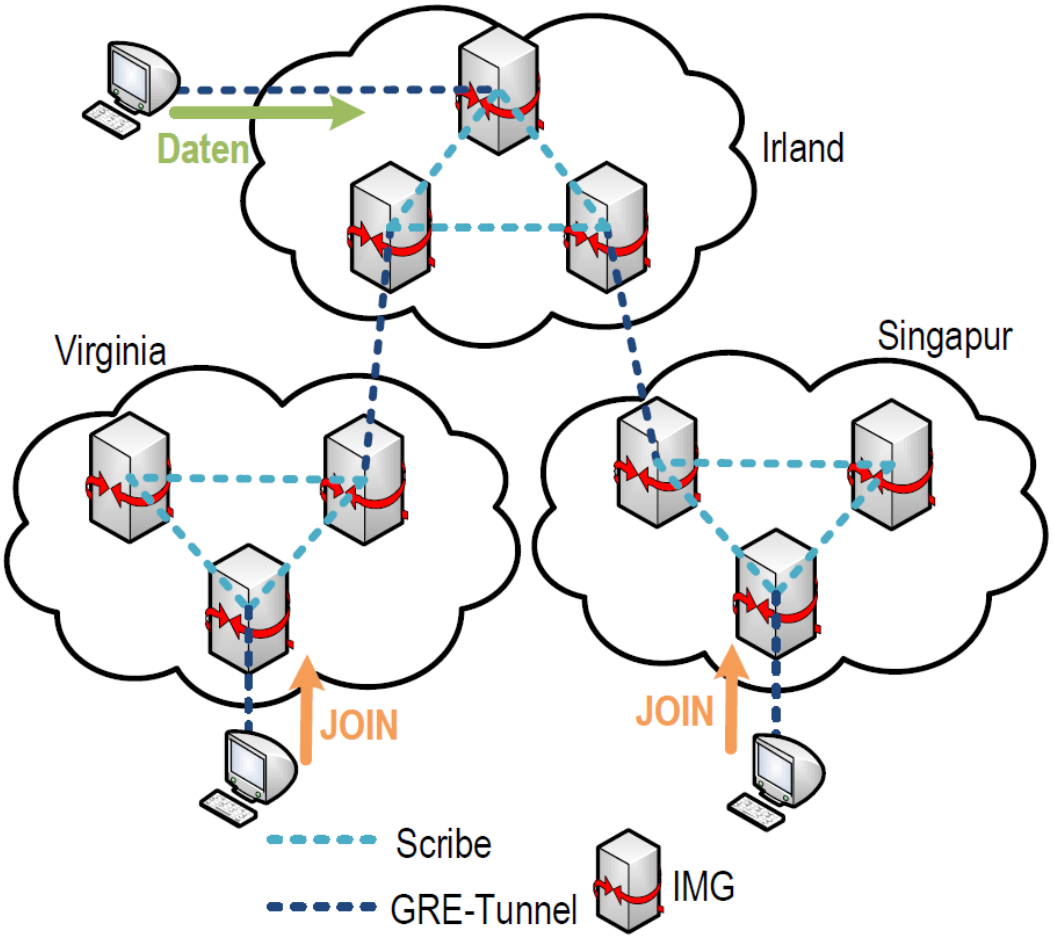

Abbildung 17: Testaufbau des Multicast-Netzwerkes

6 Evaluierung

In diesem Kapitel wird die Evaluierung des in 5.6 beschriebenen Testzenarios durchgeführt. Dazu werden Analysen zum lokalen System und zur Cloud gemacht. Es werden dabei Punkte zur Leistungsfähigkeit und zur Kostenentwicklung betrachtet.

6.1 Bandbreitenbedarf des lokalen Systems

Zunächst soll der Bandbreitenbedarf des lokalen Systems, welches Daten versendet, gezeigt werden. Beim Unicast erhöht sich dieser bei steigender Empfängerzahl und beim Multicast ist er konstant. Die Abbildung 18 zeigt den Bandbreitenbedarf pro Sekunde in Bezug auf die Empfängerzahl für einen 1MB-Datenstrom.
Damit die erhebliche Reduzierung der Last auf dem lokalen System durch Multicast möglich ist, wird der Datenstrom mittels Multicast-Tunneling an eine Instanz in der Cloud geschickt. Das Multicast-Tunneling verursacht durch die Paketkapselung eine erhöhte Paketgröße und somit einen höheren Bandbreitenbedarf. Beim GRE-Tunnel sind dies 24 Bytes und bei einem UDP-basierten Tunneling wie AMT sind es 30 Bytes. Ein Vergleich des Bandbreitenbedarfs liefert Tabelle 6. Es werden Datenstromgrößen von 500 KB bis 1500 KB aufgelistet, unter der Annahme, dass diese mit 1000 Bytes pro Paket gesendet werden. Die Erhöhung ist mit maximal 30 KB bei einem 1MB-Datenstrom gering und sollte daher keine unerwarteten Probleme hervorrufen.

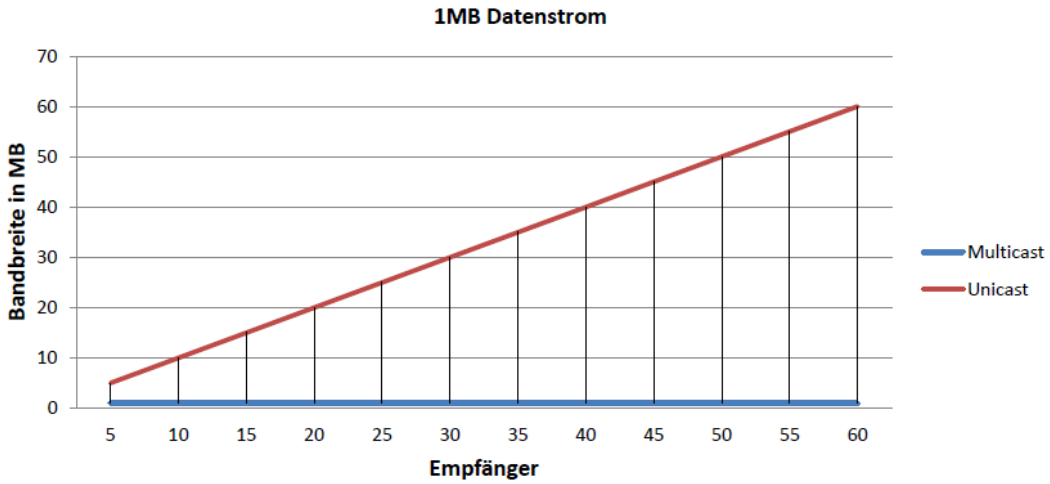

Abbildung 18: Vergleich von Unicast und Multicast

6.2 Leistung der verschiedenen Instanztypen

Die Instanztypen(siehe Tabelle 2) sorgen für die unterschiedliche Ressourcenausstattung der Instanz. Dieses wirkt sich zum Teil erheblich auf die Leistungsfähigkeit aus. Hier ist es

IP-Multicast	GRE	AMT
500	512	515
600	614	618
700	716	721
800	819	824
900	921	927
1000	1024	1030
1100	1126	1133
1200	1228	1236
1300	1331	1339
1400	1433	1442
1500	1536	1545

Tabelle 6: Bandbreiterhöhung durch Tunneling

wichtig, dass die Instanz eine hohe Netzwerkleistung hat. Es gibt keine genauen Angaben zur Netzwerkperformance der verschiedenen Instanztypen. Mit dem Netzwerkprogramm *iperf* [21] wurden einfache Messungen durchgeführt. Es wurde jeweils eine Instanz in Irland und Virginia gestartet. Dadurch soll die Netzwerkleistung aus dem Rechenzentrum hinaus gezeigt werden. Zwischen den Instanzen wurde dann ein TCP-Test mit der Fenstergröße 16 KByte durchgeführt und mehere UDP-Test bis kein Paketverlust mehr auftrat. Die Tabelle 7 zeigt die Ergebnisse für vier Instanztypen. Dieses sind grobe Richtwerte, um die Leistung einer Instanz und die Leistungsunterschiede zwischen den Instanztypen einzuordnen. Für genauere Messwerte müssten umfangreiche Tests durchgeführt werden. Dabei würde das Verhalten einer Instanz beim Versand eines Datenstromes für einen längeren Zeitraum bei steigender Empfängerzahl untersucht werden.

Instanztyp	TCP	UDP
Micro	3 MB/s	6 MB/s
Small	6 MB/s	12 MB/s
Medium	24 MB/s	24 MB/s
Large	24 MB/s	24 MB/s

Tabelle 7: Netzwerkperformance (iperf) für Micro, Small, Medium und Large

6.3 Verteilung der Instanzen

Die Instanzen werden auf mehrere entfernte Standorte verteilt, um die Daten möglichst nahe am Empfänger bereitzustellen. Der behandelte Beispielfall ist ein Sender aus der HAW-Hamburg und ein Empfänger aus einem PlanetLab-Knoten der *University of Florida*. Es gibt vier Systeme bestehend aus dem HAW-Knoten, Instanz in Irland, Instanz in Virginia

und dem PlanetLab-Knoten. Zwischen diesen Systemen wurden mittels *traceroute* die Paketlaufzeit gemessen (siehe Tabelle 8). Die Paketlaufzeit ist durch den Weg über Irland und Virgina mit 188 ms höher als der direkte Weg mit 149 ms. Auch der Weg nur über Irland ist mit der Paketlaufzeit 181 ms höher. Außerdem kommt noch die Verarbeitungszeit auf den Instanzen hinzu. Daraus lässt sich erstmal entnehmen, dass kein zeitlicher Vorteil durch das Cloud Netzwerk entsteht und das die Regionen mit 97 ms nicht besonders miteinander verbunden sind. Der Vorteil entsteht an der Stelle, dass die Verbindung des Empfängers zur Instanz in Virginia besteht und das zwischen diesen die Paketlaufzeit nur 46 ms beträgt. Die weitere Besonderheit kommt bei mehreren Empfängern aus den USA zum tragen. Durch den eingesetzten Multicast müssen die Pakete nur einmal den langen Weg von der HAW nach Virginia mit 142 ms nehmen. Ab Virginia können die Pakete mit 46 ms an die vielen Empfänger verteilt werden. Dieses ist von außen betrachtet ressourcenschonend, da die gleichen Pakete nicht mehrmals über lange Wege transportiert werden müssen. Allerdings entstehen für den Betreiber des Multicast-Netzwerkes durch die Übertragung zwischen den Regionen zusätzliche Kosten. Diese würden nicht anfallen, wenn die Pakete direkt aus Irland an die Empfänger gesendet werden.

	HAW	**Irland**	**Virginia**	**PlanetLab**
HAW	-	45 ms	109 ms	149 ms
Irland	45 ms	-	97 ms	136 ms
Virginia	109 ms	97 ms	-	46 ms
PlanetLab	149 ms	136 ms	46 ms	-

Tabelle 8: Paketlaufzeiten der vier Systeme

6.4 Kosten der Verteilung eines Datenstromes

Im Folgenden werden die Kosten für den Fall berechnet, dass ein 1-MB-Datenstrom an 90 Empfänger gesendet wird. Dabei gibt es in jeder Region 30 Empfänger verteilt auf jeweils 3 Instanzen. Der Sender hat sich zur Region Irland verbunden. Zunächst wird der gesamte Datenverkehr für Datenübertragung an 30 Empfänger abhängig von der Zeitdauer berechnet (siehe Tabelle 9). In diesem Fall müssen pro Sekunde 30 MB versendet werden, wodurch nach einigen Minuten schon mehrere Gigabytes an Datenverkehr zusammenkommen. Dieser kommt in jeder der drei Regionen zustande. In Tabelle 10 sind die dafür anfallenden Kosten aufgelistet. Für die Regionen Irland und Virginia wird der Preis von $0,120 pro GB und für die Region Singapur $0,190 pro GB genommen (siehe Tabelle 5). Der geringfügig höhere Preis pro GB in Singapur macht sich nach einen gewissen Datenvolumen stark bemerkbar. Der Datenstrom muss auch zwischen den Regionen übertragen werden, somit entsteht auch ein regionaler Datenverkehr (siehe Tabelle 11). Aufgrund des Multicast ist die Übertragung des Datenstromes nur einmal notwendig. Es ist zu beachten, dass regionaler Datenverkehr auf beiden Seiten verechnet wird. In Tabelle 12 sind die Kosten zwischen Ir-

land und Virginia und zwischen Irland und Singapur aufgeführt. Am Ende entstehen noch Kosten für die insgesamt 9 Instanzen. Die Tabelle 13 zeigt die Kosten für 3 Instanzen pro Region abhängig von der Zeit. Die jeweiligen Preise wurden aus Tabelle 4 entnommen. Wird sich an die groben Messungen aus 6.2 gehalten, dann kommen für diese Szenario nur die Instanztypen Medium oder Large zum Einsatz. Beim Instanztyp Small wird es kritisch und der Typ Micro ist eindeutig zu schwach. Insgesamt lässt sich sagen, dass bei der Verteilung eines wohlgemerkt sehr großen Datenstromes einiges an Kosten entsteht. Besonders der hohe Datenverkehr schlägt ab einer gewissen Zeit schwer ins Gewicht.

Zeitdauer	**Datenverkehr**
15 Min	27 GB
30 Min	54 GB
45 Min	81 GB
60 Min	108 GB
75 Min	135 GB
90 Min	162 GB
105 Min	189 GB
120 Min	216 GB
135 Min	243 GB
150 Min	270 GB
165 Min	297 GB
180 Min	324 GB

Tabelle 9: Datenverkehr eines 1-MB-Datenstromes für 30 Empfänger

Daten	**Irland**	**Virginia**	**Singapur**	**Gesamt**
27 GB	$3,24	$3,24	$5,13	$11,61
54 GB	$6,48	$6,48	$10,26	$23,22
81 GB	$9,72	$9,72	$15,39	$34,83
108 GB	$12,96	$12,96	$20,52	$46,44
135 GB	$16,2	$16,2	$25,65	$58,05
162 GB	$19,44	$19,44	$30,78	$69,66
189 GB	$22,68	$22,68	$35,91	$81,27
216 GB	$25,92	$25,92	$41,04	$92,88
243 GB	$29,16	$29,16	$46,17	$104,49
270 GB	$32,4	$32,4	$51,3	$116,1
297 GB	$35,64	$35,64	$56,43	$127,71
324 GB	$38,88	$38,88	$61,56	$139,32

Tabelle 10: Kosten für den Datenverbrauch

Zeitdauer	Datenverkehr
15 Min	0.9 GB
30 Min	1.8 GB
45 Min	2.7 GB
60 Min	3.6 GB
75 Min	4.5 GB
90 Min	5.4 GB
105 Min	6.3 GB
120 Min	7.2 GB
135 Min	8.1 GB
150 Min	9 GB
165 Min	9.9 GB
180 Min	10.8 GB

Tabelle 11: Regionaler Datenverkehr eines 1-MB-Datenstromes

Daten	IRL-VIR	IRL-SIN	Gesamt
0.9 GB	$0,22	$0,28	$0,50
1.8 GB	$0,43	$0,56	$0,99
2.7 GB	$0,65	$0,84	$1,49
3.6 GB	$0,86	$1,12	$1,98
4.5 GB	$1,08	$1,40	$2,48
5.4 GB	$1,30	$1,67	$2,97
6.3 GB	$1,51	$1,95	$3,47
7.2 GB	$1,73	$2,23	$3,96
8.1 GB	$1,94	$2,51	$4,46
9 GB	$2,16	$2,79	$4,95
9.9 GB	$2,38	$3,07	$5,45
10.8 GB	$2,59	$3,35	$5,94

Tabelle 12: Kosten des Regionalen Datenverkehrs

Zeit	Typ	Irland	Virginia	Singapur	Gesamt
1 Std	Micro	$0,06	$0,06	$0,06	$0,18
1 Std	Small	$0,26	$0,24	$0,26	$0,75
1 Std	Medium	$0,51	$0,48	$0,51	$1,50
1 Std	Large	$1,02	$0,96	$1,02	$3,00
2 Std	Micro	$0,12	$0,12	$0,12	$0,36
2 Std	Small	$0,51	$0,48	$0,51	$1,50
2 Std	Medium	$1,02	$0,96	$1,02	$3,00
2 Std	Large	$2,04	$1,92	$2,04	$6,00
3 Std	Micro	$0,18	$0,18	$0,18	$0,54
3 Std	Small	$0,77	$0,72	$0,77	$2,25
3 Std	Medium	$1,53	$1,44	$1,53	$4,50
3 Std	Large	$3,06	$2,88	$3,06	$9,00

Tabelle 13: Kosten für 3 Instanzen in der Region

6.5 Auswertung

Die angebotenen Dienste des Cloud Computing Anbieters Amazon ermöglichen den Aufbau eines leistungsstarken Multicast-Backbone. Sender von Multicast-Daten können diesen nutzen, um viele Empfänger zu erreichen. Endanwendern bietet sich die Möglichkeit einen Datenstrom zu empfangen. Es gilt allerdings passende Anwendungsgebiete zu schaffen. Der Aufbau des Multicast-Netzwerkes für die Verteilung eines Datenstromes zeigt den Nutzen nur bedingt. Ein Multicast-Netzwerk lebt von mehreren Sender und vielen Empfängern, welche verschiedene Gruppendaten bereitstellen und abonieren. Daher sollte ein Anbieter die Infrastruktur eines Multicast-Netzwerkes für viele Endanwender bereitstellen. Dieser übernimmt die komplette Verwaltung. Mehrere Content-Anbieter können darüber ihre Daten verteilen und die Verbraucher können zwischen verschiedenen Daten wählen.
Beim Multicast werden generell viele Daten übertragen. Im Falle von Amazon EC2 entstehen dabei nicht unerhebliche Kosten. Es wurde gezeigt, dass durch den Einsatz mehrere Standorte, die Daten intelligent verteilt und Paketlaufzeit zu den Endanwendern verkürzt werden kann. In dem umgesetzten Szenario würden nur drei Standorte eingesetzt, welches die Verteilung einschränkt. Insgesamt könnten das Multicast-Netzwerk über sieben Standorte durch Amazon EC2 gespannt werden. Interessanter für deutsche Endanwender wäre aber ein Multicast-Netzwerk über mehrere Rechenzentren in Deutschland.
H∀Mcast eignet sich als grundlegendes Softwaresystem, um die Infrastruktur des Multicast-Netzwerkes umzusetzen. Für den Einsatz in Amazon EC2 wäre aber ein Technologiemodul für einen Overlay-Multicast hilfreich, welches mit privaten und öffentlichen IP-Adressen arbeiten kann. Dadurch würde der Aufbau vereinfacht werden, denn es müssten keine Tunnel zwischen den Regionen eingerichtet werden. Ein weiterer Punkt ist das Multicast-Tunneling zu den Empfängern. Dieses wurde im bisherigen Szenario zu Testzwecken auch mit GRE-Tunneln umgesetzt. Dieses ist aber bei vielen und auch wechselnden Empfängern völlig un-

geeignet und auch nicht umsetzbar. In diesem Fall würde sich ein Modul anbieten, welches das AMT-Konzept umsetzt. Die Multicast-Pakete werden dann durch ein UDP-Kapselung an die Empfänger gesendet. Dieses Modul könnte dann dynamisch die Kommunikation zu mehreren Empfängern übernehmen. Insgesamt lässt sich sagen, dass das aufgebaute System einsatzfähig ist, aber das dieses für einen produktiven Einsatz mit vielen Teilnehmern an vielen Stellen erweitert werden muss.

7 Zusammenfassung und Ausblick

Das Ziel dieser Arbeit war die Entwicklung und Evaluierung eines adaptiven hybriden Multicast auf Basis von Cloud Networking. Dabei wurden die Kernthemen Multicast und Cloud Computing untersucht. Die Vorteile des Multicast bei der Datenverteilung, aber auch dessen Verfügbarkeitsproblem wurde gezeigt. Es wurde vermittelt, dass ein Cloud Computing Anbieter Rechen -und Netzwerkressourcen sofort auf Anfrage bereitstellen kann. Durch den Einsatz der Cloud Computing Ressourcen sollte versucht werden die Verfügbarkeitsprobleme des Multicast zu überbrücken, indem ein leistungsstaker Multicast-Backbone innerhalb der Cloud Computing Umgebung aufgebaut wird. Das Softwaresystem für den adaptiven hybriden Multicast lieferte H∀Mcast. Der genutzte Cloud Computing Anbieter für die Rechen -und Netzwerkressourcen war Amazon EC2. Die Software H∀Mcast wurde in Form eines Amazon Machine Images bereitgestellt. Multicast-Tunneling und Overlay-Multicast dienten zur Lösung des nicht gewährleisteten IP-Multicast. Die Verteilung erfolgte auf die drei Regionen Irland, Virginia und Singapur. Die Implementierung lief darauf hinaus, dass ein bestimmtes Testszenario umgesetzt wird. Dabei wurde ein Multicast-Netzwerk über die drei Regionen mit jeweils drei Instanzen aufgebaut. Anhand diesem wurden kleinere Messungen durchgeführt und die Kostenrechnung für die Verteilung eines 1-MB-Datenstromes aufgelistet.

Letztendlich wurde ein Grundsystem geschaffen, mit welchen es möglich ist, Multicast-Kommunikation durch die Ressourcen eines Cloud Computing Anbieters über entfernte Standorte stattfinden zu lassen. Allerdings muss dieses für einen produktiven Einsatz an vielen Stellen überarbeitet werden. Dazu zählt die Umsetzung des dynamischen Multicast-Tunnelings und der Overlay-Multicast über Regionsgrenzen hinweg, sollte keine direktes Tunneling zwischen den Regionen erwünscht sein. Desweiteren können genauere Messungen zu der Leistungsfähigkeit einer Instanz unter Einsatz der H∀Mcast-Middleware durchgeführt werden. Es kann sich auch nach anderen Cloud Computing Anbietern umgesehen werden, welche beispielweise Rechenzentren in Deutschland oder auch andere Preise für Datenübertragungen anbieten. Am Ende gilt es auch konkrete Anwendungsgebiete zu schaffen, sei es die Verteilung von Audiostreams, um das Gesamtsystem zu vermarkten und viele Teilnehmer zu bekommen.

Literatur

[1] Cisco Visual Networking Index: Forecast and Methodology, 2010–2015. Technical report, Cisco Systems, 2011.

[2] Unlocking Video over the Internet with MX Series Routers. Juniper Networks, 2011.

[3] Amazon Web Services. *Overview of Amazon Web Services*, 2010.

[4] Amazon Web Services. Amazon EC2 User Guide. Website, 2012. Available online at `http://aws.amazon.com/documentation/ec2/`; visited on May 30th 2012.

[5] Amazon Web Services. Amazon Elastic Compute Cloud. Website, 2012. Available online at `http://aws.amazon.com/ec2/`; visited on May 30th 2012.

[6] Amazon Web Services. AWS SDK for Java. Website, 2012. Available online at `http://aws.amazon.com/sdkforjava/`; visited on May 18th 2012.

[7] Amazon Web Services. VPC FAQs. Website, 2012. Available online at `http://aws.amazon.com/vpc/faqs/#R4`; visited on May 18th 2012.

[8] S. Bhattacharyya. An Overview of Source-Specific Multicast (SSM). RFC 3569, IETF, July 2003.

[9] Gregory Bumgardner. Automatic Multicast Tunneling. Internet-Draft – work in progress 13, IETF, April 2012.

[10] B. Cain, S. Deering, I. Kouvelas, B. Fenner, and A. Thyagarajan. Internet Group Management Protocol, Version 3. RFC 3376, IETF, October 2002.

[11] Miguel Castro, Peter Druschel, Anne-Marie Kermarrec, and Antony Rowstron. SCRIBE: A large-scale and decentralized application-level multicast infrastructure. *IEEE Journal on Selected Areas in Communications*, 20(8):100–110, 2002.

[12] Steve Deering. Host extensions for IP multicasting. RFC 1112, IETF, August 1989.

[13] Dino Farinacci, Tony Li, Stan Hanks, David Meyer, and Paul Traina. Generic Routing Encapsulation (GRE). RFC 2784, IETF, March 2000.

[14] B. Fenner, M. Handley, H. Holbrook, and I. Kouvelas. Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised). RFC 4601, IETF, August 2006.

[15] INET research group Hamburg. HAMcast - Developers Webpages. Website, 2012. Available online at `http://hamcast.realmv6.org/developers`; visited on May 18th 2012.

[16] Xing Jin, Kan-Leung Cheng, and S.-H. Gary Chan. Island multicast: combining IP multicast with overlay data distribution. *IEEE Transactions on Multimedia*, 11(5):1024–1036, 2009.

[17] Shaofei Lu, Jianxin Wang, Guanzhong Yang, and Chao Guo. SHM: Scalable and Backbone Topology-Aware Hybrid Multicast. In *16th Intern. Conf. on Computer Communications and Networks (ICCCN'07)*, pages 699–703, August 2007.

[18] Sebastian Meiling, Dominik Charousset, Thomas C. Schmidt, and Matthias Wählisch. System-assisted Service Evolution for a Future Internet – The HAMcast Approach to Pervasive Multicast. In *Proc. of IEEE GLOBECOM 2010, Workshop MCS 2010*, pages 913–917, Piscataway, NJ, USA, Dec. 2010. IEEE Press.

[19] Sebastian Meiling, Dominik Charousset, Thomas C. Schmidt, and Matthias Wählisch. HAMcast – Evaluierung einer systemzentrierten Middleware-Komponente für einen universellen Multicast-Dienst im Future Internet. *Praxis der Informationsverarbeitung und Kommunikation (PIK)*, 2012.

[20] Peter Mell and Timothy Grance. The NIST Definition of Cloud Computing. Technical report, National Institute of Standards and Technology, 2011.

[21] NLANR/DAST. iperf. Website, 2008. Available online at `http://sourceforge.net/projects/iperf/`; visited on May 30th 2012.

[22] J. Postel. User Datagram Protocol. RFC 768, IETF, August 1980.

[23] Sylvia Ratnasamy, Mark Handley, Richard M. Karp, and Scott Shenker. Application-Level Multicast Using Content-Addressable Networks. In Jon Crowcroft and Markus Hofmann, editors, *Proc. of 3rd Intern. Workshop on Network Group Communication (NGC'01)*, volume 2233 of *LNCS*, pages 14–29, London, UK, Nov. 2001. Springer–Verlag.

[24] Theodore John Socolofsky and Claudia Jeanne Kale. TCP/IP tutorial. RFC 1180, IETF, January 1991.

[25] R. Vida and L. Costa. Multicast Listener Discovery Version 2 (MLDv2) for IPv6. RFC 3810, IETF, June 2004.

[26] Matthias Waehlisch, Stig Venaas, and Thomas Schmidt. A Common API for Transparent Hybrid Multicast. Internet-Draft – work in progress 04, IETF, January 2012.

[27] Matthias Wählisch and Thomas C. Schmidt. Between Underlay and Overlay: On Deployable, Efficient, Mobility-agnostic Group Communication Services. *Internet Research*, 17(5):519–534, November 2007.

[28] Matthias Wählisch, Thomas C. Schmidt, and Georg Wittenburg. BIDIR-SAM: Large-Scale Content Distribution in Structured Overlay Networks. In Mohamed Younis and Chun Tung Chou, editors, *Proc. of the 34th IEEE Conference on Local Computer Networks (LCN)*, pages 372–375, Piscataway, NJ, USA, October 2009. IEEE Press.

[29] D. Waitzman, C. Partridge, and S. Deering. Distance Vector Multicast Routing Protocol. RFC 1075, IETF, November 1988.

[30] Beichuan Zhang, Wenjie Wang, Sugih Jamin, Daniel Massey, and Lixia Zhang. Universal IP multicast delivery. *Computer Networks*, 50(6):781–806, 2006.

*Versicherung über Selbstständigkeit

Hiermit versichere ich, dass ich die vorliegende Arbeit im Sinne der Prüfungsordnung nach §22(4) ohne fremde Hilfe selbstständig verfasst und nur die angegebenen Hilfsmittel benutzt habe.

X l X Hamburg, 19. Juni 2012

Ort, Datum Unterschrift