Middleware – Cloud Computing

Einführung

Wintersemester 2020/21

Tobias Distler

Friedrich-Alexander-Universität Erlangen-Nürnberg
Lehrstuhl Informatik 4 (Verteilte Systeme und Betriebssysteme)

Lehrstuhl für Verteilte Systeme und Betriebssysteme

FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Überblick

Einführung
- Überblick
- Herausforderungen

Cloud Computing

- Merkmale
  - **Auslagerung von Diensten**, Berechnungen und/oder Daten
  - Verfügbarkeit **scheinbar unbegrenzter Ressourcen**
  - Einfacher universeller Zugriff
  - **Schnelle dynamische Skalierbarkeit**
- Grundlagen
  - Hochskalierbare verteilte Infrastrukturen auf Provider-Seite
  - Leistungsfähige Netzwerkanbindung auf Client-Seite
- Literatur

Macha Creeger
Cloud Computing: An Overview
Queue – Distributed Computing, 7(5), 2009.

Michael Armbrust, Armando Fox, Rean Griffith, Anthony D. Joseph, Randy Katz et al.
A View of Cloud Computing
Communications of the ACM, 53(4):50–58, 2010.

Skalierbarkeit

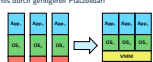
- Häufiges Problem: Auslastungsabhängige Bereitstellung von Ressourcen für Dienste
  - Lastentwicklung eventuell unbekannt
  - **Ungünstiges Verhältnis zwischen Spitzen- und Durchschnittslast**
  - Starke Lastschwankungen über den Tag bzw. das Jahr hinweg
- Mögliche **Konsequenzen ungenauer Bedarfsvorhersagen**
  - Bereitstellung von zu wenigen Ressourcen (*Underprovisioning*)
  - Bereitstellung von zu vielen Ressourcen (*Overprovisioning*)
- Potentielle Vorteile durch Verlagerung von Diensten in die Cloud
  - Verfügbarkeit zusätzlicher Ressourcen im Sekunden- bzw. Minutenbereich
  - **Dynamische Skalierbarkeit in beide Richtungen**
  - Abrechnungsmodell: *Pay-as-you-go*
    - Kosten orientieren sich am tatsächlichen Ressourcenverbrauch
    - Feingranulare Abrechnung [Beispiele: Virtuelle Maschine: pro Stunde; Netzwerk: pro Megabyte]
    - **Achtung:** Dienste in der Cloud zu betreiben ist nicht automatisch günstiger!

Verfügbarkeit

- Wartung und Reparatur von Systemkomponenten
  - Aufgabe des Cloud-Anbieters
  - **Einschränkung von Verfügbarkeitsgarantien** für Cloud-Dienste
  - Nutzer hat keinen Einfluss auf Zeitpunkt und Dauer der Maßnahmen
- Technische Infrastruktur in Cloud-Datenzentren
  - Zusammenschluss einer großen Anzahl verhältnismäßig kleiner Server
  - Günstige Einkaufspreise aufgrund großer Stückzahlen
  - Konsequenzen
    - **Ausfälle einzelner Komponenten werden zum Regelfall**
    - Kompatibilitätsprobleme aufgrund heterogener Hardware
- Realistisches Fehlerszenario: Ausfall kompletter Datenzentren
- Maßnahmen zur **Tolerierung von Fehlern**
  - Verteilung eines Dienstes auf verschiedene Datenzentren
  - Replikation von Daten über mehrere Standorte

Basistechnologien

- **Web-Services**
  - Sprachunabhängige Basis für entfernte Kommunikation
  - Bereitstellung von Diensten in der Cloud
  - Schnittstelle zur Cloud-Konfigurierung
- **Virtualisierung**
  - Paralleler Betrieb mehrerer *virtueller Maschinen* auf einem Rechner
  - Höhere Auslastung einzelner Rechner [0.8% (ohne Virt.) → bis zu 80% (mit Virt.) [Creeger]]
  - Kostenersparnis durch geringeren Platzbedarf



Everything as a Service

- Kategorien
  - **Software as a Service (SaaS)**
    - Bereitstellung von Endnutzer verwendeter Dienste
    - Beispiel: Google Docs
  - **Platform as a Service (PaaS)**
    - Bereitstellung von Middleware zur Implementierung komplexer Dienste
    - Beispiel: Google AppEngine
  - **Infrastructure as a Service (IaaS)**
    - Bereitstellung von Rechen- und Speicherinfrastruktur
    - Beispiel: Amazon EC2
- In der Praxis
  - Oftmals als Schichten aufeinander aufbauend
  - Grenzen zwischen Kategorien fließend

Einsatzszenarien

- **Öffentliche Cloud** (*Public Cloud*)
  - Unternehmen (z. B. Amazon, Microsoft) stellen ihre Infrastruktur zur Verfügung
  - Cloud-Nutzer müssen selbst vergleichsweise wenige Ressourcen vorhalten
- **Private Cloud**
  - Nutzung der bereits im eigenen Unternehmen vorhandenen Infrastruktur
  - Einsatz von Virtualisierung zur flexiblen Verwaltung von Ressourcen
- **Hybride Cloud**
  - Kombination aus privater und öffentlicher Cloud
  - Mögliche Aufteilung
    - Kritische Daten verbleiben im privaten Teil der Cloud
    - Öffentliche Cloud vor allem zur Deckung von Bedarfsspitzen
- **Multi Clouds / Cloud-of-Clouds**
  - Parallele Nutzung verschiedener öffentlicher Clouds
  - Absicherung gegen den Ausfall eines Cloud-Anbieters

Limitierungen und offene Fragen

- „*Vendor Lock-In*"-Problem: Starke Abhängigkeit von einem Cloud-Anbieter
  - **Erschwerter Anbieterwechsel**
  - Gründe: fehlende Standards, aufwendiger Datentransfer
- Technische Limitierungen
  - **Ineffizienter Transfer großer Datenmengen** in die bzw. aus der Cloud
    [Amazon bietet dafür z. B. an, Daten per Festplatte zu transferieren: https://aws.amazon.com/de/snowball/]
  - Optimale Isolation von virtuellen Maschinen ist nicht immer möglich
    - Sicherheitsprobleme (z. B. Schwachstellen in der Virtualisierungssoftware)
    - Problem der *Performance Isolation*: Instabile bzw. unvorhersehbare Performanz bestimmter Operationen (z. B. Festplattenzugriffe)
- Weiterführende Aspekte
  - **Vertraulichkeit von Daten**
  - Rechtliche Fragen (Beispiele)
    - Dürfen medizinische Daten in einer öffentlichen Cloud verarbeitet werden?
    - Werden gesetzliche Bestimmungen zum Speicherort von Daten eingehalten?

Überblick

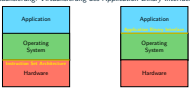
Einführung

Überblick

Herausforderungen

Wie lässt sich Virtualisierung praktikabel realisieren?

- **Anforderungen** an ein virtualisiertes System
  - Äquivalenz
  - Ressourcenkontrolle
  - Effizienz
- **Virtualisierungsebenen**
  - Systemvirtualisierung: Virtualisierung der *Instruction Set Architecture*
  - Prozessvirtualisierung: Virtualisierung des *Application Binary Interface*



Wie wird die eigene Infrastruktur für andere nutzbar?

- **Aufbau einer Infrastruktur-Cloud**

- **Aufgabenbereiche**
  - Verwaltung von physischen Maschinen
  - Verwaltung und Platzierung von virtuellen Maschinen
  - Anbindung an Datenspeicher

Wie lassen sich große Datenmengen verwalten?

- Ansatz
  - Speziell auf die jeweiligen Anforderungen zugeschnittene Systeme
  - **Enge Verzahnung mit der Anwendung**
- **Beispiel: Google File System**
  - Anforderungen
    - Sehr große Dateien
    - Hauptsächlich sequentielle Schreibzugriffe, kaum Modifikationen
  - Kein Dateisystem im klassischen Sinne
  - Optimierte Auslastung der Netzwerkverbindungen
- **Beispiel: Amazon Dynamo**
  - Anforderungen
    - Große Anzahl an vergleichsweise kleinen Datensätzen
    - Hohe Verfügbarkeit
  - Replizierter Datenspeicher für Schlüssel-Wert-Paare
  - Abgeschwächte Konsistenzgarantien

Wie lassen sich große Datenmengen verarbeiten?

- Beispiel: Google (und viele andere)
  - Anforderungen
    - **Parallele Nutzung einer großen Anzahl von Rechnern**
    - Einfache Realisierung von Anwendungen
  - *MapReduce*
    - Framework übernimmt Verteilung der Anwendung
    - Programmierer implementiert **zwei Methoden**
      - Map: Abbildung der Eingabedaten auf Schlüssel-Wert-Paare
      - Reduce: Zusammenführung der von Map erzeugten Schlüssel-Wert-Paare
- Koordinierung und Konfigurierung verteilter Anwendungen
  - Anforderungen
    - **Abstimmung zwischen einer großen Anzahl von Prozessen**
    - Ausfallsichere Verwaltung von Konfigurationsinformationen
  - Beispiel: *Chubby* (Google)
    - Bereitstellung als externer Koordinierungsdienst
    - Generische Schnittstelle zur Implementierung komplexer Abstraktionen