Grid, Cloud und Peer to Peer

Hochleistungs-Ein-/Ausgabe

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Michael Kuhn

2019-06-04

Wissenschaftliches Rechnen
Fachbereich Informatik
Universität Hamburg

Grid, Cloud und Peer to Peer

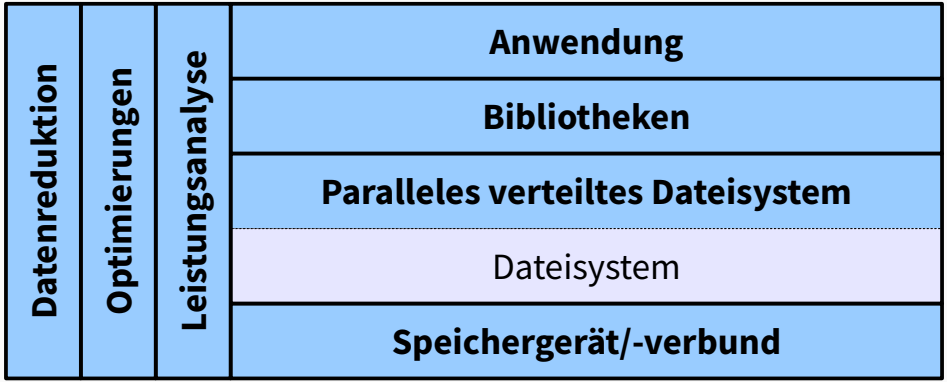


Abbildung 1: E/A-Schichten und orthogonale Themen

Motivation

- Hohe Leistungsfähigkeit und Speicherkapazität immer häufiger nötig
  - Komplexe Simulationen, Big-Data-Analysen etc.
- Nicht immer ist ein Supercomputer vor Ort verfügbar
  - Oder der Computer vor Ort ist alleine nicht leistungsfähig genug
- Verschiedene Konzepte zur Nutzung fremder Ressourcen
  - Konkret Grid, Cloud und Peer to Peer

Grid

- Rechenleistung wie Strom aus der Steckdose
  - Überall und für alle verfügbar
  - Name inspiriert vom “power grid”
- Unterschiede zu Hochleistungsrechnern
  - Geographisch weiter verteilt
  - Heterogenere Architekturen
  - Häufig ein Cluster von Clustern
- Hauptsächlich im wissenschaftlichen Umfeld

Cloud

- Rechenleistung und Speicher wie Strom aus der Steckdose
  - Fortsetzung des Grid-Gedankens
- Einfacher zugänglich
  - Häufig über Webschnittstellen steuerbar
- Dynamische Struktur
  - Skalierung durch Hinzunehmen zusätzlicher Ressourcen
- Deutlich weitere Verbreitung
  - Webentwicklung, Backup etc.

- Koordination Gleichgestellter (Peers)
  - Im Gegensatz zu Client-Server-Systemen
- Viele Einsatzgebiete
  - Streaming, Bitcoin etc.
  - Hauptsächlich Bereitstellung von Daten
- Sehr weite Verbreitung
  - $\approx$ 10 % des weltweiten Datentransfers

Überblick

- Üblicherweise andere Probleme als im HPC
  - Häufig keine Hochleistungsvernetzung etc.
  - Viele unabhängige Berechnungen
- Heterogenität der beteiligten Ressourcen
  - Betriebssystem und Bibliotheken
  - Latenz und Durchsatz
- Authentifizierung ist wichtiger Aspekt
  - Grid-weite Identität
  - Geregelt über Zertifikate etc.

Überblick... [1]

1. *"... coordinates resources that are not subject to centralized control ..."*
   - Ressourcen unterschiedlicher Institutionen
   - Unterschiedliche Ressourcentypen (Desktop vs. Server)
2. *"... using standard, open, general-purpose protocols and interfaces ..."*
   - Authentifizierung und Autorisierung
   - Standardisierter Zugriff auf Ressourcen
3. *"... to deliver nontrivial qualities of service."*
   - Dienstgüte je nach Benutzeranforderungen
   - Mehr als die Summe der Teile

Organisation

- Koordinierte Nutzung verfügbarer Ressourcen
  - Gemeinsame Lösung von Problemen
- Virtuelle Organisationen
  - Dynamisch und institutsübergreifend
- Vereinbarung über Ressourcennutzung
  - Alle Partner sollten etwas beisteuern
- Ziel ist ein globales, einheitliches Grid

Organisation...

- Programm benötigt hohe Leistung
  - Programme üblicherweise nicht interaktiv
    - Wie im Hochleistungsrechnen
- Ausführung wird über Scheduler geregelt
  - Ähnlich wie im Hochleistungsrechnen
    - Allerdings zusätzliche Rahmenbedingungen
- Programm und Daten müssen zusammengeführt werden
  - Programm kann auf entferntem Rechner ausgeführt werden
  - Daten liegen allerdings lokal und müssen übertragen werden

Probleme

- Programm wird auf Rechner A ausgeführt
  - Transferzeit: Sekunden bis Minuten
  - Eventuell noch Kompilieren für Zielarchitektur
- Daten befinden sich auf Rechner B
  - Annahme: Rechner sind mit Gbit-Ethernet verbunden
  - Transferzeit: 450 GB/h bei 125 MB/s
- Nach Berechnung eventuell Transfer der Ergebnisse
  - Möglicherweise im TB- bis PB-Bereich

Probleme...

- Neben Durchsatz auch Kapazität ein Problem
  - Daten sind möglicherweise sehr groß
- Originaldaten befinden sich auf Rechner A
  - Rechner A hat großes Speichersystem
- Daten werden auf Rechner B transferiert
  - Temporäre Speicherung belegt Platz
  - Ausreichend Platz für Ergebnisdaten notwendig

Probleme...

- Vertraulichkeit der Daten
  - Daten unter Umständen (zeitweise) vertraulich
  - Veröffentlichung erst nach erster Publikation
  - Im eigenen Rechenzentrum relativ einfach zu gewährleisten
- Keine Informationen über entfernte Rechnerumgebung
  - Datensicherheit, Zugangskontrolle etc.
  - Möglicherweise konkurrierende Benutzer
- Stark abhängig von der Art der Daten
  - Militärische Forschung oder Ergebnisse eines Milliardenprojekts

Typen

- Rechen-Grids (Computing)
  - Rechenkapazität aus der Steckdose
- Daten-Grids (Data)
  - Datenbestände werden verknüpft
  - Zugang häufig über Grid-Portal
- Ressourcen-Grids (Resource)
- Dienstleistungs-Grids (Service)
- Wissens-Grids (Knowledge)

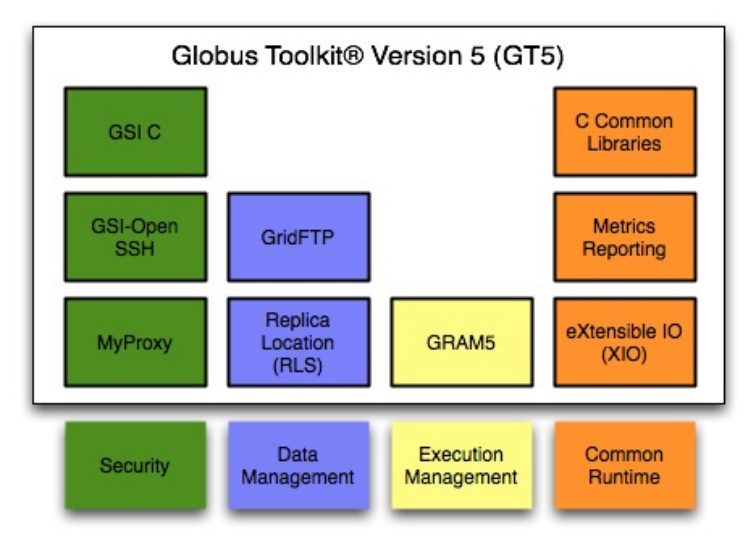
- Beispiele: C3-Grid (Collaborative Climate Community Data and Processing Grid), ESGF (Earth System Grid Federation)

Architektur

- Open Grid Services Architecture (OGSA)
  - Vorläufer: Open Grid Services Infrastructure (OGSI)
- Beteiligte Ressourcen sind Grid-Dienste
  - Inzwischen in Form von Web-Services
- Middleware
  - g-Eclipse, Globus Toolkit, Unicore, gLite, Sun Grid Engine etc.
  - *“The nice thing about standards is that you have so many to choose from.”*
    – Andrew S. Tanenbaum

- Sammlung unterschiedlicher Komponenten
  - Grid Resource Allocation & Management Protocol (GRAM)
  - Monitoring and Discovery Service (MDS)
  - Grid Security Infrastructure (GSI)
  - Global Access to Secondary Storage (GASS) und GridFTP
- Bietet ein Fundament für Grid
  - Satz von Komponenten zur Entwicklung eigener Software
  - Kompatibilität zwischen unterschiedlichen Institutionen

Globus... [5]


Globus Toolkit® Version 5 (GT5)
GSI C
C Common Libraries
GSI-Open SSH
GridFTP
Metrics Reporting
MyProxy
Replica Location (RLS)
GRAM5
eXtensible IO (XIO)
Security
Data Management
Execution Management
Common Runtime

- Benötigt ein Zertifikat der virtuellen Institution
  - Teilweise recht aufwendig
    - Vorlegen des Personalausweises etc.
- Danach Erzeugung eines temporären Proxys
  - Aufruf: `grid-proxy-init`
  - Status anzeigen mit `grid-proxy-info`
- Danach Datentransfer möglich
  - Aufruf: `globus-url-copy source destination`

- Unterstützung für mehrere Protokolle
  - `file://`, `ftp://`, `gsiftp://`, `http://` und `https://`
- Unterstützung für mehrere parallele Datenströme
  - Manchmal notwendig, um ausreichend Durchsatz zu erreichen
- Verglichen mit normalen FTP oder SCP relativ kompliziert

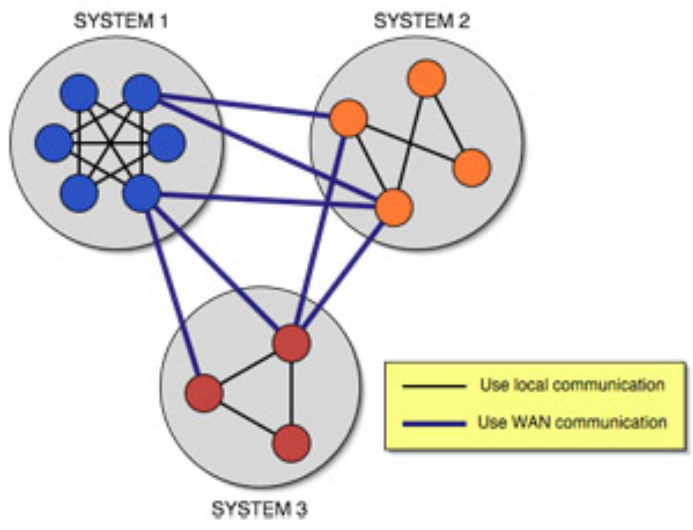
```
1 $ grid-proxy-init -valid 1:00
2 $ grid-proxy-info
3 $ globus-url-copy -parallel 4 file:///home/user/file
      ↪ gsiftp://server.example.com/tmp/
4 $ globus-url-copy -list gsiftp://server.example.com/tmp/
```

Listing 1: GridFTP-Beispiel

- Setzt üblicherweise auf normalen Dateisystemen auf
  - Erweiterung von FTP
  - Unterstützung von Grid-Authentifizierung
- Kompatibilität zwischen verschiedenen Institutionen
  - Immer noch institutionsspezifische Pfade etc.
  - Jobscripte müssen unter Umständen angepasst werden
  - Bekanntes Problem im Hochleistungsrechnen
    - Üblicherweise mehrere Konfigurationen für verschiedene Hochleistungsrechner

MPICH-G2 [2]

- MPICH mit Grid-Unterstützung
  - Veraltet, basiert auf MPI 1.1
- Grid-Techniken für bessere Integration
  - Starten von Prozessen auf entfernten Systemen
  - Staging von Programmen und Daten
  - Sicherheit
- Automatische Auswahl der Kommunikationsmethode
  - Hochleistungsvernetzung innerhalb des Clusters
  - IP zwischen Clustern


SYSTEM 1
SYSTEM 2
SYSTEM 3
Use local communication
Use WAN communication

Überblick

- Ähnliches Konzept wie Grid
  - Berechnung und Daten „in der Wolke“
- Keine genaue Kenntnis über Ressourcen notwendig
  - Automatische Ausführung auf verfügbaren Ressourcen
- Im Gegensatz zu Grid zentralisierter Ansatz
  - Anbieter kontrolliert Ressourcen
- Populär im kommerziellen Sektor
  - Amazon, Google, Microsoft, Backblaze etc.

Everything as a Service [3]

- Infrastructure as a Service (IaaS)
  - Zugriff auf (virtualisierte) Hardware
  - Eigenes Betriebssystem etc.
- Platform as a Service (PaaS)
  - Durch Cloud-Anbieter definierte Plattform
  - Erlaubt Anwendungen darauf zu entwickeln
- Software as a Service (SaaS)
  - Zugriff auf Software
  - Auch Software on Demand

Everything as a Service...

- `/dev/null` as a Service ☺
  - 25 GB pro Monat kostenlos
  - “We support BigData!”
  - “Run huge Map-Reduce jobs on the data you won’t see anymore!”
  - `LD_PRELOAD`-Bibliothek für transparente Nutzung
  - Webseite: `https://devnull-as-a-service.com/`

Liefermodelle [3]

- Public Cloud
  - Öffentlich zugänglich
  - Üblicherweise verbrauchsabhängige Bezahlung
- Private Cloud
  - Infrastruktur innerhalb der eigenen Organisation
- Hybrid Cloud
  - Eine Kombination aus Public und Private Cloud
- Community Cloud
  - Wie bei Public Cloud, allerdings kleinerer Nutzerkreis
  - Beispiel: Sciebo (Campuscloud)

Charakteristika [3]

1. *“On-demand self-service”*
   - Benutzer können automatisiert Ressourcen anfordern
   - Keine menschliche Interaktion notwendig
2. *“Broad network access”*
   - Verfügbarkeit über das Netzwerk
   - Zugang über Standardmechanismen und unterschiedliche Plattformen
3. *“Resource pooling”*
   - Ressourcen befinden sich in einem Pool und können von mehreren Benutzern in Anspruch genommen werden
   - Dynamische Zuteilung nach Bedarf

Charakteristika... [3]

4. *“Rapid elasticity”*
   - Ressourcen können nach Bedarf dynamisch skaliert werden
   - Verfügbare Ressourcen erscheinen unlimitiert
5. *“Measured service”*
   - Ressourcen werden automatisiert kontrolliert und optimiert
   - Benutzung kann überwacht und gemeldet werden

- Charakteristika auch für das Hochleistungsrechnen interessant
  - Selbst-provisionierbare Dateisysteme
  - Elastische Dateisysteme

Daten

- Daten sind kein so großes Problem wie bei Grid
  - Datentransfer über große Entfernungen problematisch
- Berechnung und Daten oft beim selben Anbieter
  - Keine Migration notwendig
  - Üblicherweise gute Anbindung
    - Teilweise mit garantiertem Durchsatz
- Häufig kein normales Dateisystem
  - Stattdessen Objektspeicher
  - Zugriff oft über HTTP

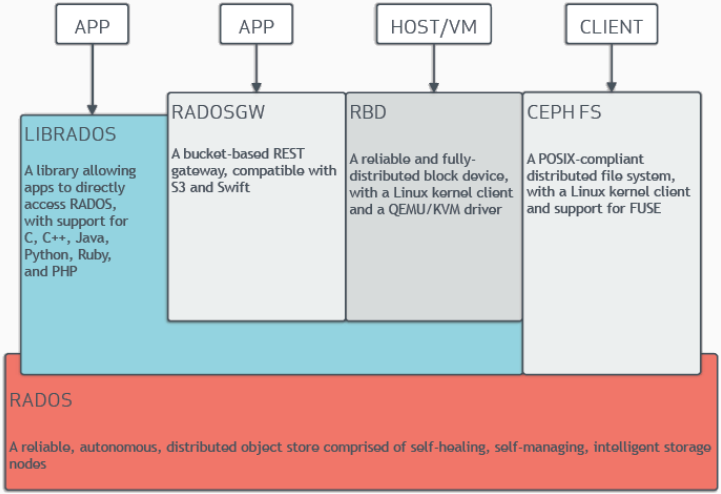
- Amazon Simple Storage Service (S3) sehr beliebt
  - Teil der Amazon Web Services (AWS)
  - reddit, Dropbox, Minecraft, Tumblr etc.
- S3-Schnittstelle ist ein häufig verwendeter Standard
  - Google Cloud Storage
  - OpenStack Swift
  - Ceph mit RADOS-Gateway

- Inzwischen auch Cloud-HPC
  - Früher Fokus auf Komfort
- Amazon Elastic Compute Cloud (EC2)
  - C5 und C5n-Instanzen für das Hochleistungsrechnen
  - Intel Xeon Platinum mit Zugriff auf Intel AVX-512, Intel Turbo Boost und Enhanced Networking
    - Optional mit knoten-lokalen NVMe-SSDs
  - Optimierte Anbindung
    - Bis zu 25 Gbit/s Netzwerkdurchsatz
    - Bis zu 14 Gbit/s für Elastic Block Storage (EBS)
  - Unterstützung für das Erstellen von Clustern

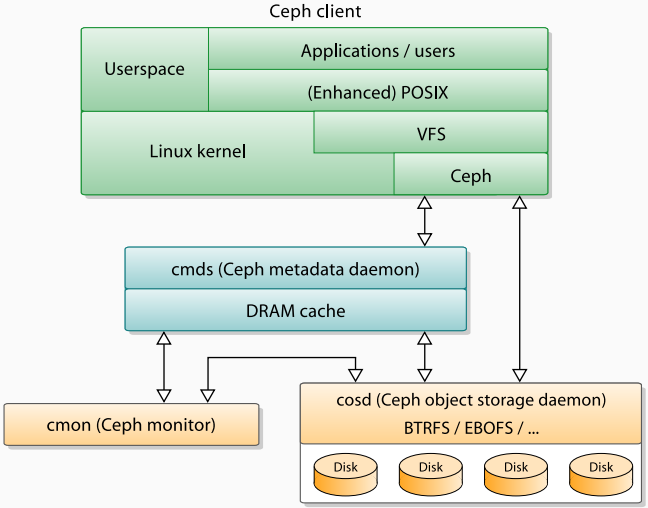
- Überlegung: Was kostet ein Supercomputer in der Cloud?
  - Beispiel: DKRZ, Mistral
  - Ca. 3.000 Knoten mit jeweils 30 Kernen (Durchschnitt)
- Entspricht ungefähr c5.9xlarge mit 36 Kernen
  - $\approx$ 0,75 $/h bei Laufzeit von 3 Jahren und Vorauszahlung
  - Entspricht 2.250 $/h, 54.000 $/d und 19.710.000 $/a
  - 98.550.000 $ bei einer Laufzeit von 5 Jahren ($\approx$ 87.956.565 €)
  - 229.950.000 $ ($\approx$ 205.231.985 €) bei On-Demand-Instanzen
- Vergleich: Kosten für Mistral
  - 40.000.000 € Anschaffung
  - 2.000.000 €/a Betrieb
  - 50.000.000 € bei einer Laufzeit von 5 Jahren

- Etwas weniger Arbeitsspeicher
  - 72 GiB pro Instanz (insgesamt 216 TB)
  - Mistral insgesamt 240 TB
- Außerdem noch keine Speicherkosten enthalten
  - Mistral: Lustre-Dateisystem mit 60 PB und Bandarchiv mit 200 PB
- Dateisystem über Elastic Block Storage
  - 0,054 $/GB pro Monat, insgesamt 3.240.000 $ pro Monat, 38.880.000 $/a
  - 194.400.000 $ für 5 Jahre (≈ 173.503.361 €)
- Archiv über Glacier
  - 0,0045 $/GB pro Monat, insgesamt 900.000 $ pro Monat, 10.800.000 $/a
  - 54.000.000 $ für 5 Jahre (≈ 48.195.378 €)

- Ceph ist eine Speicherplattform
  - Bietet Datei-, Objekt- und Blockspeicher
  - Kein Single Point of Failure
  - Skalierbar bis in den Exabyte-Bereich
  - Fehlertoleranz durch Replikation
- Kein Cloud-System, wird aber häufig als Basis verwendet
  - S3-kompatible Schnittstelle

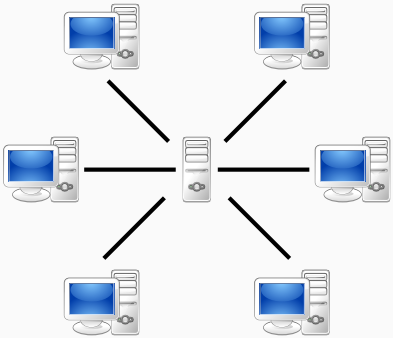

APP
APP
HOST/VM
CLIENT
LIBRADOS
A library allowing apps to directly access RADOS, with support for C, C++, Java, Python, Ruby, and PHP
RADOSGW
A bucket-based REST gateway, compatible with S3 and Swift
RBD
A reliable and fully-distributed block device, with a Linux kernel client and a QEMU/KVM driver
CEPH FS
A POSIX-compliant distributed file system, with a Linux kernel client and support for FUSE
RADOS
A reliable, autonomous, distributed object store comprised of self-healing, self-managing, intelligent storage nodes

- Basis ist ein verteilter Object Store
  - Komplette Verwaltung wird von RADOS übernommen
  - Darauf aufbauend zusätzliche Funktionalitäten
  - Oder direkter Zugriff auf den Object Store
- Verteilte Blockgeräte
  - Kann für lokale Dateisysteme genutzt werden
- POSIX-Dateisystem
  - CephFS stellt Dateisystemfunktionalität bereit

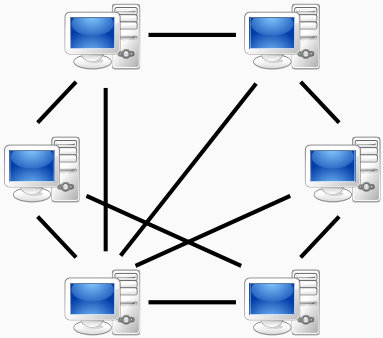

Ceph client
Userspace
Applications / users
(Enhanced) POSIX
Linux kernel
VFS
Ceph
cmds (Ceph metadata daemon)
DRAM cache
cmon (Ceph monitor)
cosd (Ceph object storage daemon)
BTRFS / EBOFS / ...
Disk
Disk
Disk
Disk

- Typische Komponenten
  - Object Storage Daemon für Daten
  - Metadata Daemon für Metadaten
- Daten werden in lokalem Dateisystem gespeichert
  - Früher EBOFS, wird nicht mehr unterstützt
  - Aktuell btrfs, dadurch POSIX auf zwei Schichten

- Peer to Peer (P2P) bekannt aus Tauschbörsen
  - Peers sind Gleichgestellte im Netzwerk
  - Im Gegensatz zu Client-Server-Systemen
- Teilnehmer können Dienste anbieten und in Anspruch nehmen
  - Üblicherweise aber Zuweisung bestimmter Dienste
  - Häufig beschränkt auf Datenaustausch
- Teilnehmer kommunizieren direkt untereinander
  - Keine zentrale Instanz, die Flaschenhals sein könnte



(a) Client-Server-System [8]



(b) Peer-to-Peer-System [8]

Probleme

- Teilnehmer sind sehr heterogen
  - Unterschiedliche Rechenleistung, Durchsatz, Latenz etc.
  - Teilnehmer können dem Netzwerk beliebig beitreten
- Verfügbarkeit der Teilnehmer ist nicht garantiert
  - Redundanz zwingend notwendig
  - Beitreten und Verlassen wird als *Churn* bezeichnet

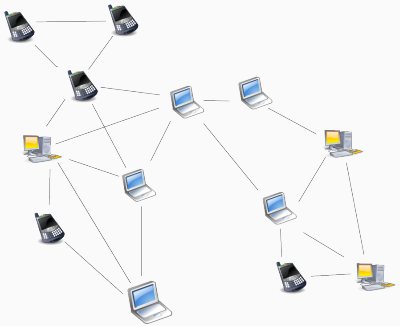
Dezentralisierung

- Keine zentrale Datenbasis
  - Jeder Teilnehmer stellt Daten und Speicherplatz bereit
  - Teilnehmer kennen nicht gesamten Datenbestand
- Keine zentrale Kontrollinstanz
  - Manchmal aber Vermittler für bessere Leistung
  - Beispiel: BitTorrent-Tracker
- Unterschiedliche Grade der Dezentralisierung

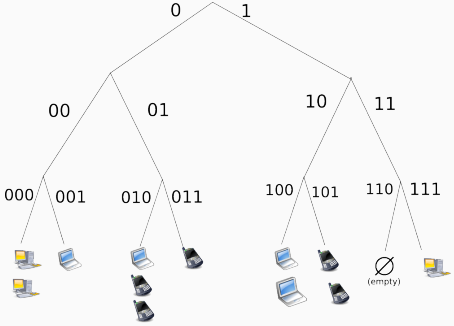
Dezentralisierung...

- Vollständig zentralisiert
  - Benötigt zentrale Instanzen, um zu funktionieren
  - Üblicherweise Verwaltung der Peers und Daten
- Hybride Zentralisierung
  - Einige Peers nehmen Sonderaufgaben wahr
    - Sogenannte Supernodes
- Vollständig dezentralisiert
  - Alle Peers nehmen dieselben Aufgaben wahr
  - Theoretisch beliebig skalierbar
  - Hohe Fehlertoleranz

- Peers bilden ein sogenanntes Overlay-Netzwerk
  - Beschreibt die Verbindungen zwischen den Teilnehmern
  - Üblicherweise unabhängig vom physikalischen Netzwerk
- Unterschiedliche Grade der Strukturierung



(c) Unstrukturiertes Overlay-Netzwerk [8]



(d) Strukturiertes Overlay-Netzwerk [8]

Strukturierung...

- Unstrukturiert
  - Einfach zu realisieren
  - Peers verbinden sich zufällig miteinander
  - Alle Peers sind gleich, kein Problem bei hohem Churn
  - Schwierig bezüglich Suche
- Strukturiert
  - Bestimmte Topologie vorgegeben
  - Meistens über eine verteilte Hashtabelle (DHT) realisiert
  - Hoher Churn problematischer durch ständige Neuorganisation der Struktur

Vergleich

- Ähnliches Modell wie bei Grid und Cloud denkbar
  - Daten sind von überall zugreifbar
  - Daten sind einigermaßen sicher
    - Hängt vom Grad der Redundanz ab
  - Keine eigene Hardware notwendig
- Anbieter für Datenhaltung bezahlen
  - Früher: Wuala
    - Daten werden verschlüsselt, aufgeteilt und an Peers verteilt
    - Konzept konnte sich nie wirklich durchsetzen

- Übliche Anwendung ist File Sharing
  - Daten werden einmal eingestellt und dann geteilt
    - Aufteilung in Blöcke für parallelen Transfer
    - Keine nachträglichen Modifikationen möglich
- Dateisysteme sind aber auch möglich
  - Teilnehmer können Daten ändern
  - Nur Forschungsprototypen verfügbar

- Ivy
  - Unterstützt mehrere Benutzer
  - Unterstützung für Lesen und Schreiben
  - Auf Basis von Logs und DHT
- Shark
  - Daten stammen von zentralem Server
  - Kooperatives Caching verteilt Last
- OceanStore
  - Starke Konsistenz durch Commit-Protokoll
  - Konsistenzanforderungen können gelockert werden

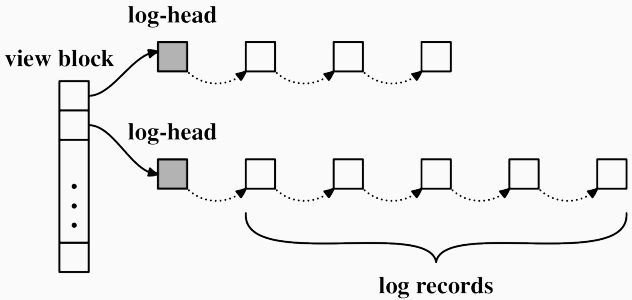


Figure 1: Example Ivy view and logs. White boxes are DHash content-hash blocks; gray boxes are public-key blocks.

- Dateisystem besteht aus mehreren Logs
  - Ein Log pro Teilnehmer, das alle Änderungen enthält
  - Einfügen in das eigene Log, Lesen aus allen Logs

Zusammenfassung

- Grid
  - Bereitstellung von Ressourcen zur entfernten Nutzung
  - Komplexe Benutzung (Zertifikate, virtuelle Organisationen)
- Cloud
  - Bereitstellung von Ressourcen zur entfernten Nutzung
  - Deutlich einfachere Handhabung (webbasiert)
- Peer to Peer
  - Bereitstellung von Informationen (meistens Dateien)
  - Üblicherweise alles öffentlich

Zusammenfassung...

- Grids, Clouds und Peer to Peer haben ähnliche Konzepte
- Berechnung ist einigermaßen einfach zu realisieren
  - Clouds sind (noch) teurer als ein eigener Hochleistungsrechner
- Daten sind problematisch
  - Daten zu bewegen ist teurer als sie zu berechnen
  - Müssen zum/vom Ort der Berechnung transportiert werden
  - Einschränkungen bezüglich Kapazität und Durchsatz
- Bei Peer to Peer unterschiedliche Grade der Dezentralisierung und Strukturierung
  - Meistens mit verteilten Hashtabellen

Quellen

[1] Ian Foster. **What is the Grid? A Three Point Checklist.** `http://www.mcs.anl.gov/~itf/Articles/WhatIsTheGrid.pdf`, 07 2002.

[2] Nicholas T. Karonis, Brian R. Toonen, and Ian T. Foster. **MPICH-G2: A grid-enabled implementation of the message passing interface.** *J. Parallel Distrib. Comput.*, 63(5):551–563, 2003.

[3] Peter M. Mell and Timothy Grance. **SP 800-145. The NIST Definition of Cloud Computing.** Technical report, Gaithersburg, MD, United States, 2011.

[4] Athicha Muthitacharoen, Robert Morris, Thomer M. Gil, and Benjie Chen. **Ivy: A Read/Write Peer-to-peer File System.** *SIGOPS Oper. Syst. Rev.*, 36(SI):31–44, December 2002.

Quellen ...

[5] University of Chicago. **Globus Toolkit.** http://toolkit.globus.org/toolkit/.

[6] University of Chicago. **MPICH-G2.** http://toolkit.globus.org/grid_software/computation/mpich-g2.php.

[7] Wikipedia. **Ceph (software).** https://en.wikipedia.org/wiki/Ceph_(software).

[8] Wikipedia. **Peer-to-peer.** https://en.wikipedia.org/wiki/Peer-to-peer.