Object Storage mit Policy-Unterstützung für AFS

Felix Frank

DESY

GUUG Treffen Berlin, 05. März 2009

HELMHOLTZ GEMEINSCHAFT

DESY

Über DESY

Deutsches Elektronensynchotron

Ein Forschungszentrum der Helmholtz-Gemeinschaft

- Standorte Hamburg und Zeuthen
- Grundlagenforschung in der Teilchenphysik

Übersicht

Das AFS Dateisystem

- Vergleich mit NFSv3
- Grundlegende AFS-Konzepte
- Alternativ: Object Storage Systeme, Beispiel Lustre

Die Vereinigung: Das OpenAFS+OSD Projekt

- Idee und Funktion
- Nützliche Features von OpenAFS+OSD
- Fileserver Policies für OSD-Verwendung

Was ist AFS?

- verteiltes Dateisystem mit globalem Namensraum

```
diff /afs/ifh.de/user/f/ffrank/volume.c
/afs/ipp-garching.mpg.de/.cs/volume.c
```

- Kerberos integriert, für WAN entwickelt

Motivation

- viel flexibler als NFSv3, NFSv4 nicht bereit
- OpenSource

Status

- OpenAFS 1.4.x stabil, 1.5.x testing und Windows-Client
- 194 Sites registriert, viele mehr unregistriert
- neue Features werden noch immer implementiert
- Pläne für Foundation

Vergleich AFS / NFSv3

	NFSv3	AFS
Setup	sehr einfach	komplex
Administrierung	OK mit automount	einfach, transparent
Wartung	Downtime	Live
max. # Clients	recht begrenzt	skaliert sehr gut
Caching (Client)	vorhanden	hochentwickelt
Einsatz im HPC	schwierig	RO Daten repliziert
Durchsatz	schnell	Client langsam
Authentisierung	UNIX UIDs	Kerberos
ACLs	nur UNIX mode	nur für Directories
Ausfallsicherheit	begrenzt	Live-Redundanz
Einbindung	einfach	Kernel Modul notw.
Windows API	nicht trivial	1.5.x stabil
Support	gr. Userbase	aktive Community

AFS: Administrationskonzepte

Volume

- Dateneinheit mit eigenem Quota
- RW + opt. 1...n RO + opt. Backup
- Mountpoints
  - Teil der Daten in einem Volume
    - Realisiert mittels Symlink
  - “User Level Automounter”
  - Client baut Verzeichnisbaum lokal auf
    - Flexibilität, Skalierbarkeit

```
$ fs mkm /afs/ifh.de/packages/RPMS p.rpm
```

Access Control Lists

- gelten stets für gesamte Verzeichnisse
- read, lookup, insert, delete, write, lock, administer
- individuell zu vergeben für Nutzer und Gruppen
- nützlich: ACLs für IP-Adressen (Rechner, Netze)

Backup Volumes

Namensschema für Volumes, z.B. Homes:

- user.ffrank
- user.hartmut
- ...

Mountpoints z.B.

/afs/ifh.de/user/f/ffrank → `user.ffrank`

...

Backups erzeugen

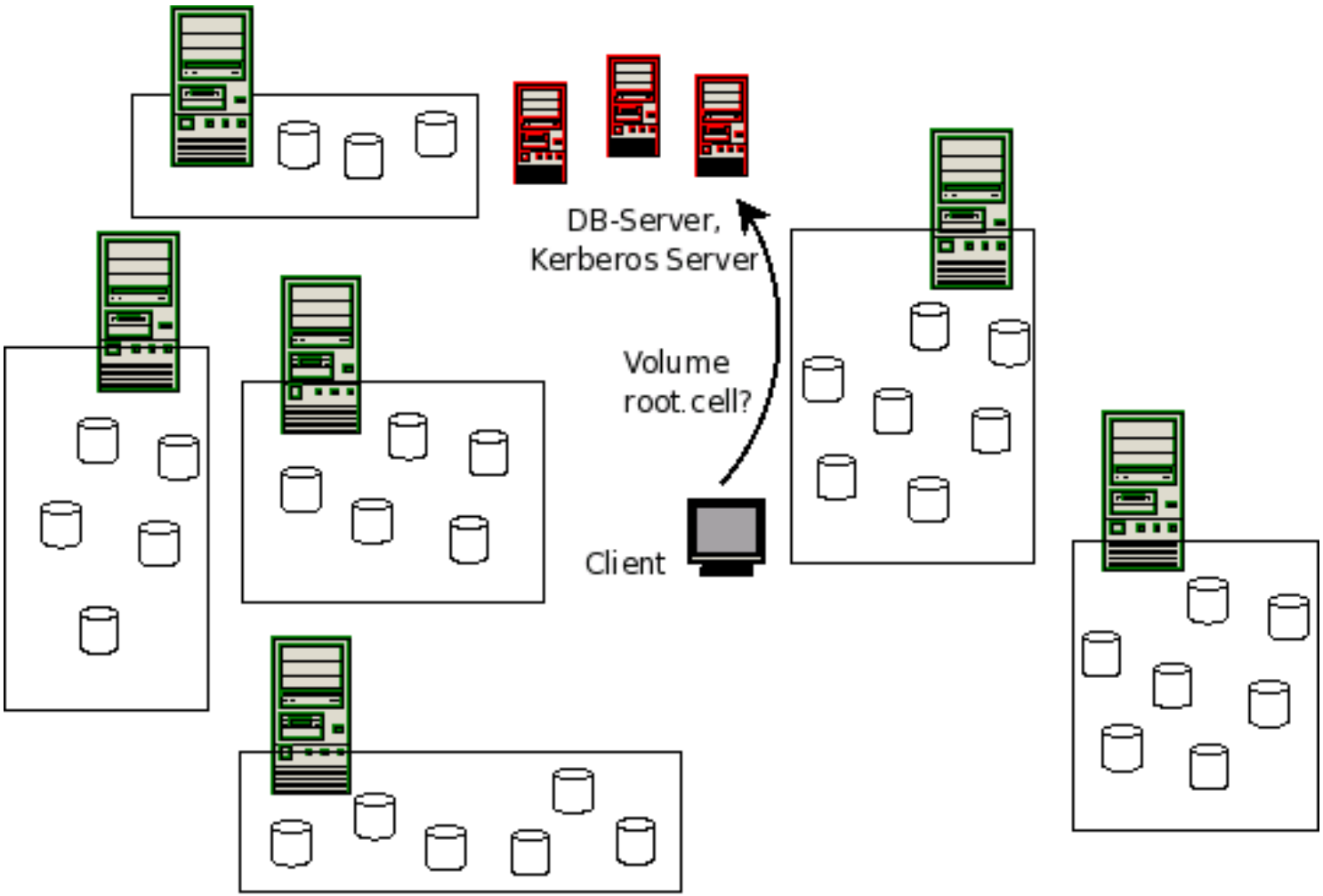
```
$ vos backupsys -prefix ``user.''
```

/afs/ifh.de/user/f/ffrank/.OldFiles

→ `user.ffrank.backup`

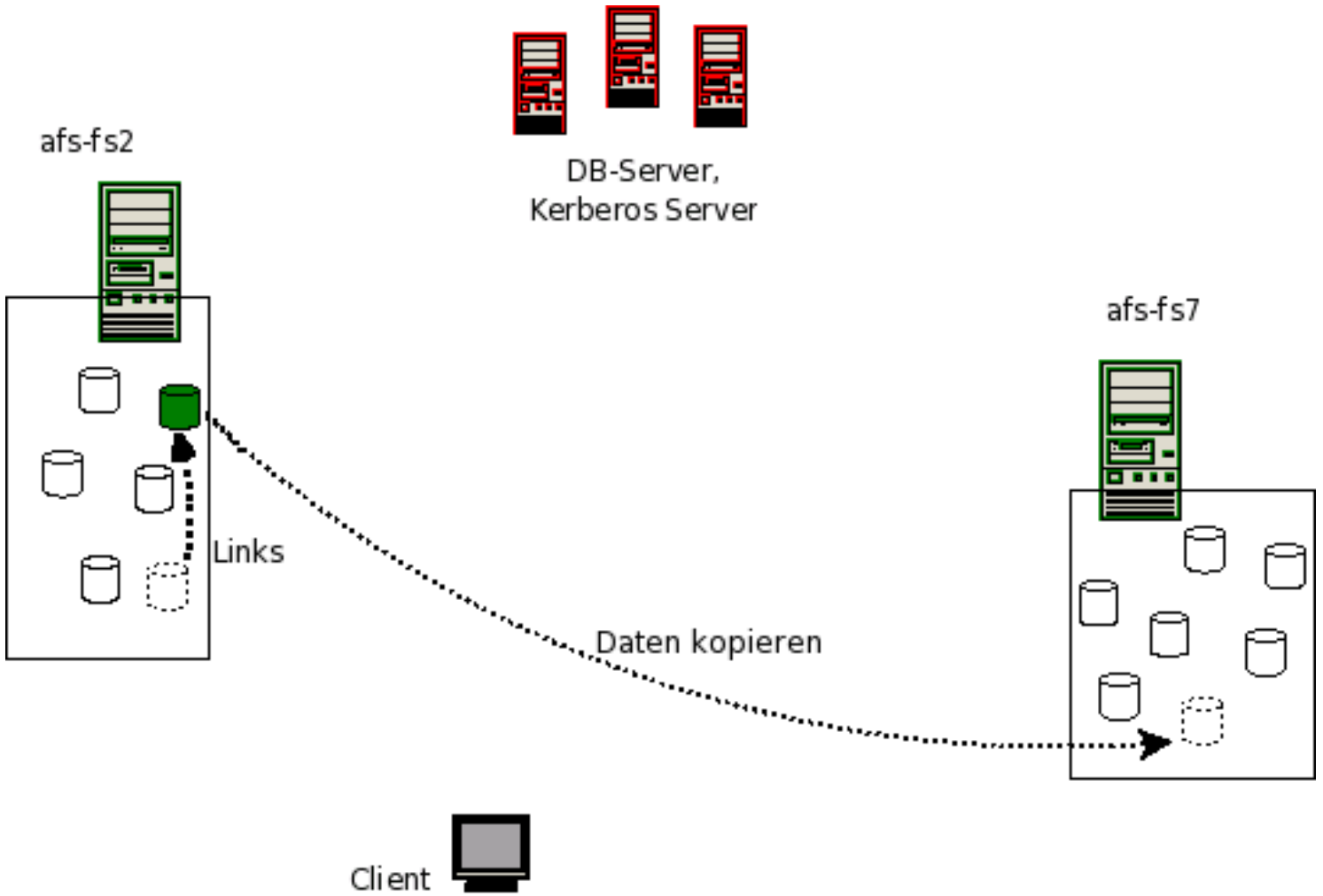
...

Übersicht: AFS Server



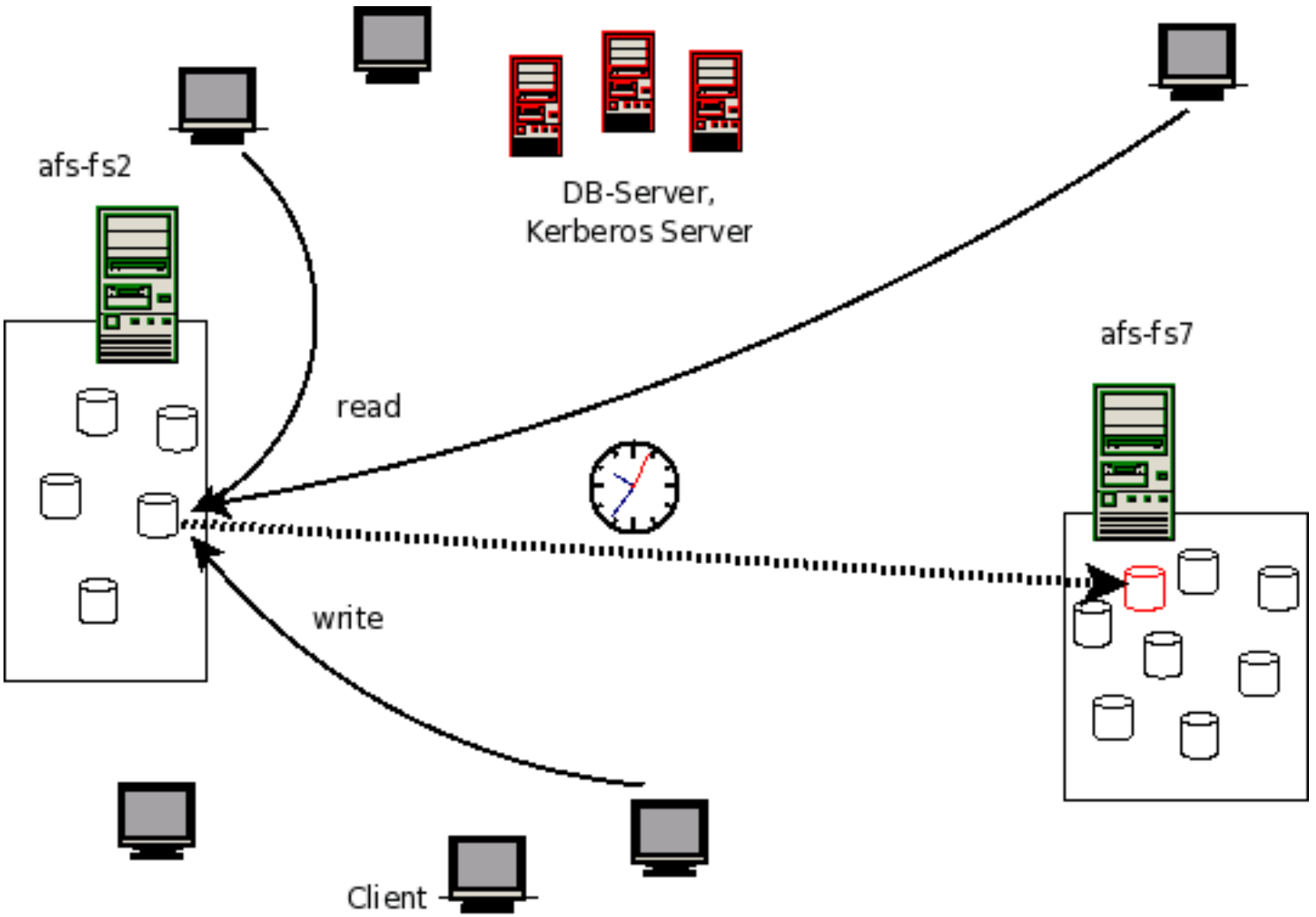
Datenreplikation

Daten werden verteilt



Datenmigration

Zugriffe funktionieren weiter



Ausblick Filesysteme – Zeuthen

NFS

- Version 3 verliert Bedeutung
- Version 4 kam zu spät, auch komplex, vieles noch TODO

AFS

- Community arbeitet an Performance
- Hotspots machen Probleme mit Farm-Computing
- fest etabliert, NFSv4 (noch) keine Alternative
- Nutzung: 24.1 / 52 TB

Lustre

- in Zeuthen Ersatz für kommerzielles **Panasas**
- schon jetzt wesentlich bessere Performance als AFS und NFS
- andere Nachteile

Übersicht

Das AFS Dateisystem

- Vergleich mit NFSv3
- Grundlegende AFS-Konzepte
- Alternativ: Object Storage Systeme, Beispiel Lustre

Die Vereinigung: Das OpenAFS+OSD Projekt

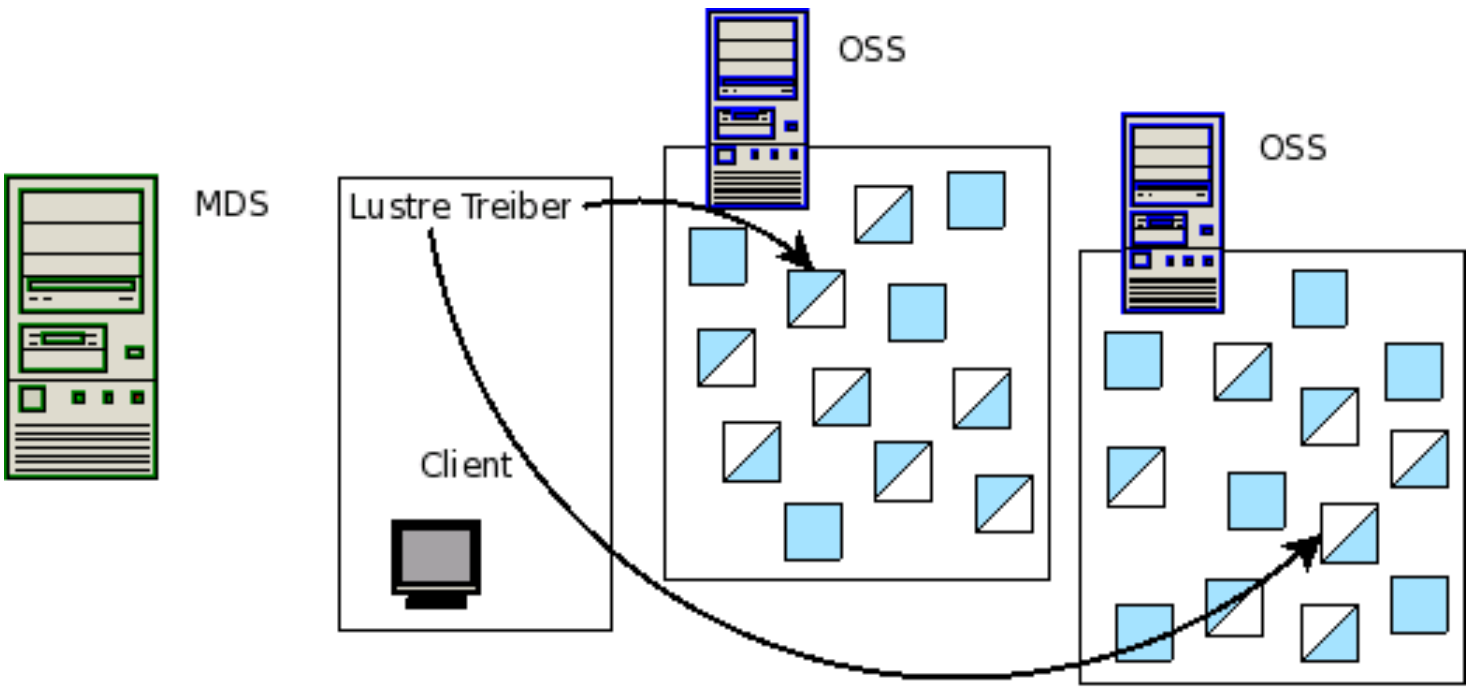
- Idee und Funktion
- Nützliche Features von OpenAFS+OSD
- Fileserver Policies für OSD-Verwendung

Object Storage – Beispiel: Lustre

- zentrale Verwaltungsdatenbank: MGS
- Dateiverzeichnis: Metadataserver
- Daten stark verteilt auf Objectstorageservern
  - halten nicht Dateien sondern *Objekte*
  - Datei kann aus mehreren Objekten bestehen

Zugriff

- Client fragt Dateiinformationen erst bei MDS an
- Verbindet sich mit OSS(n) um Daten zu empfangen



Lustre: Vor- und Nachteile

Vorteile

- Datenverkehr stets verteilt
- extrem schneller Client und Server
- unterstützt RDMA über InfiniBand

Nachteile

- bei Hochlast: Metadataserver wird Bottleneck
  - derzeit Live-Redundanz sehr beschränkt
  - Wartbarkeit bei AFS DB-Servern besser
- Storage Server nicht leicht ersetzbar
- Kernel Modul nicht portabel, gepatchter Kernel nötig (Server)
- Server nur auf Linux
- Erweiterbarkeit begrenzt
- Kerberos, ACLs, Quota etc. noch in Beta bzw. Roadmap

Übersicht verteilte Dateisysteme

NFS	**AFS**	**Lustre**
sehr einfach	Setup komplex	Setup komplex
schnell nur für einzelne Clients	Lastverteilung (readonly)	schnelles Lesen und Schreiben
max.Geschw.OK	Durchsatzlimits	schnell!
Wartung schwierig	transparente Datenmigration	keine Datenmigration
keine Redundanz in v3	redundante DB + RO-Replikation	fehlende Redundanz der DB
keine Verteilung	RW Hotspots	gute Datenverteilung

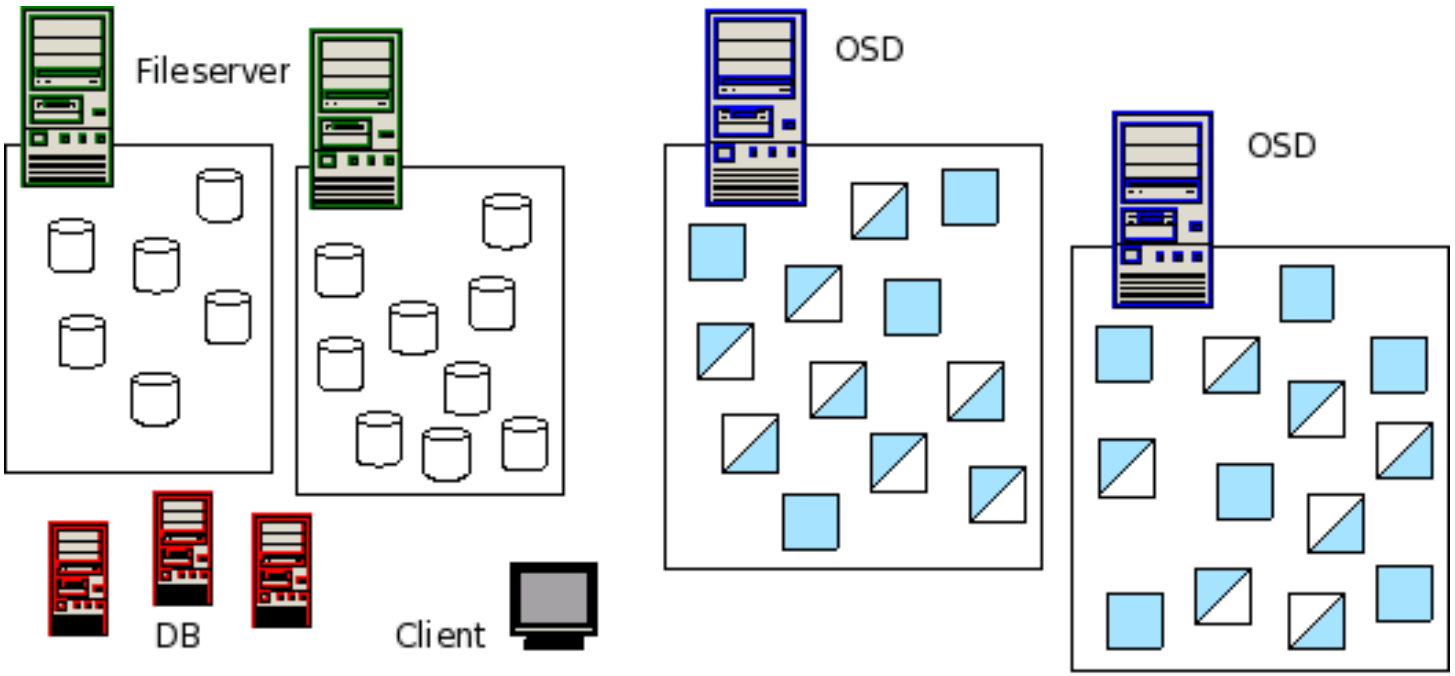
Übersicht

Das AFS Dateisystem
- Vergleich mit NFSv3
- Grundlegende AFS-Konzepte
- Alternativ: Object Storage Systeme, Beispiel Lustre

Die Vereinigung: Das OpenAFS+OSD Projekt
- Idee und Funktion
- Nützliche Features von OpenAFS+OSD
- Fileserver Policies für OSD-Verwendung

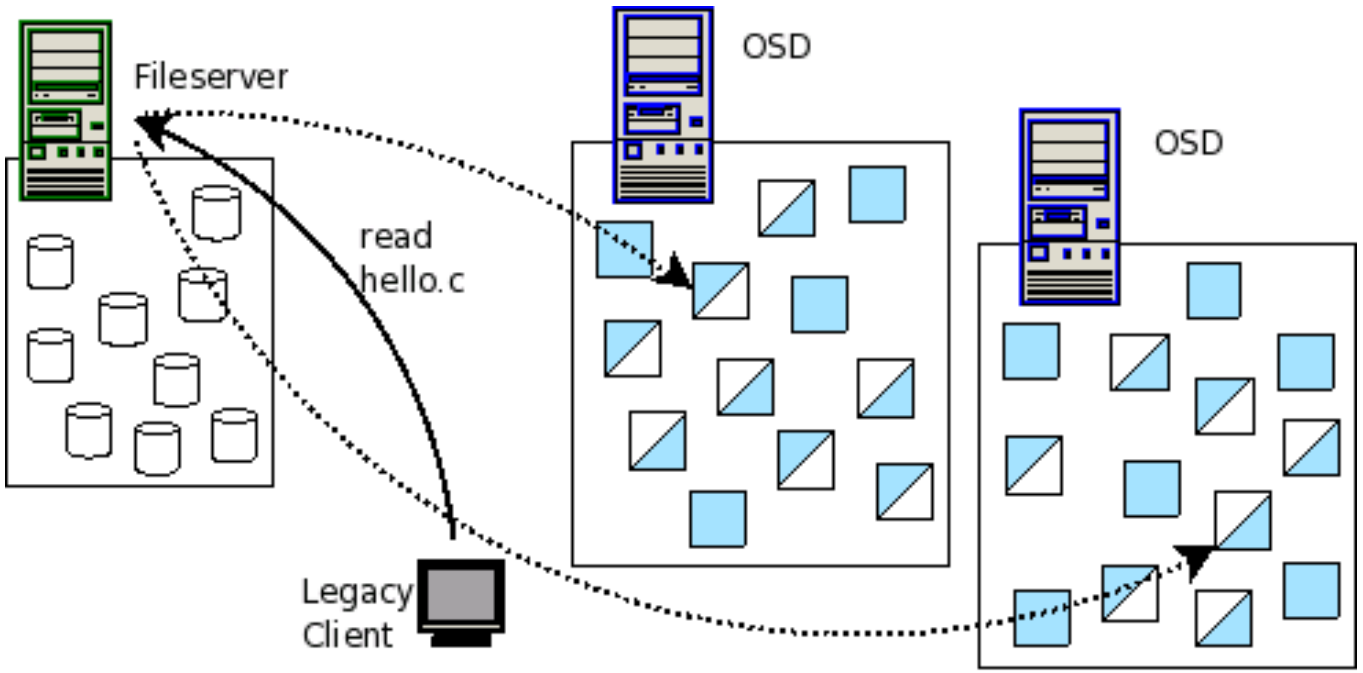
Auftritt OpenAFS+OSD

- (fast) vollständig kompatible Erweiterung von AFS
- herkömmlicher AFS Client wird unterstützt
- Object Storage Devices werden zusätzlich zu AFS Fileservern eingesetzt
- dadurch Vermeidung von Hotspots



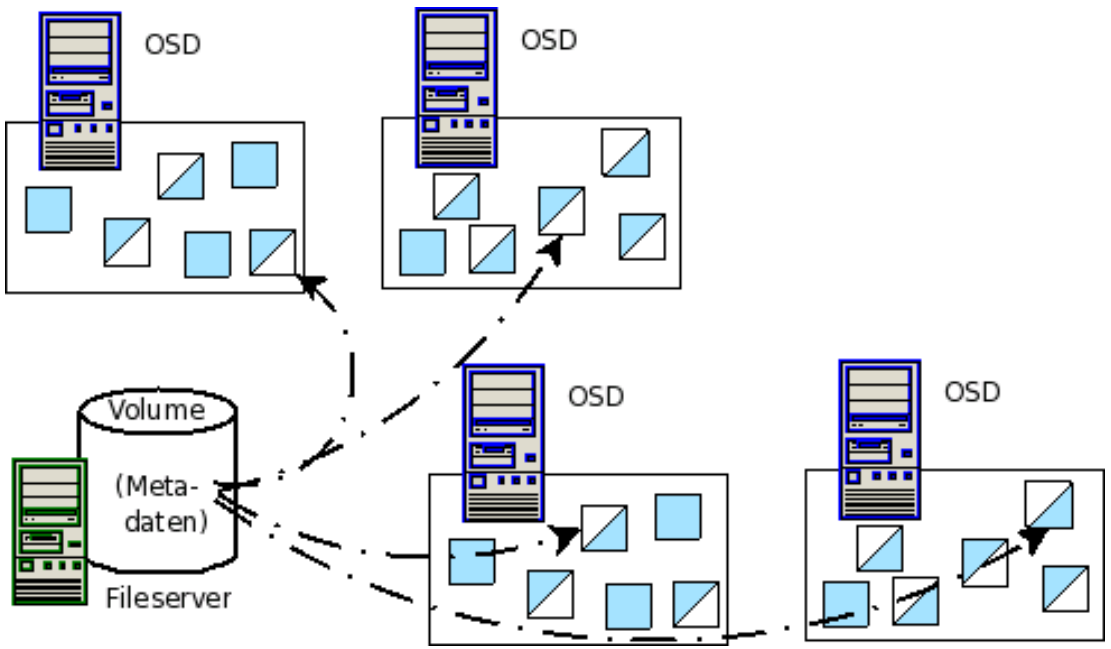
Funktionsweise von AFS+OSD

- Fileserver agiert gleichzeitig als MDS
- Daten von File sind entweder direkt im Volume oder auf beliebigem OSD
- OSD wird verwendet wenn Datei >1MB (z.B.)
- im Volume auf Fileserver dann nur Metadaten
- Client kommuniziert mit OSDs
- für “alte” Clients spricht Fileserver transparent mit OSDs



Performance: Stripes und Copies

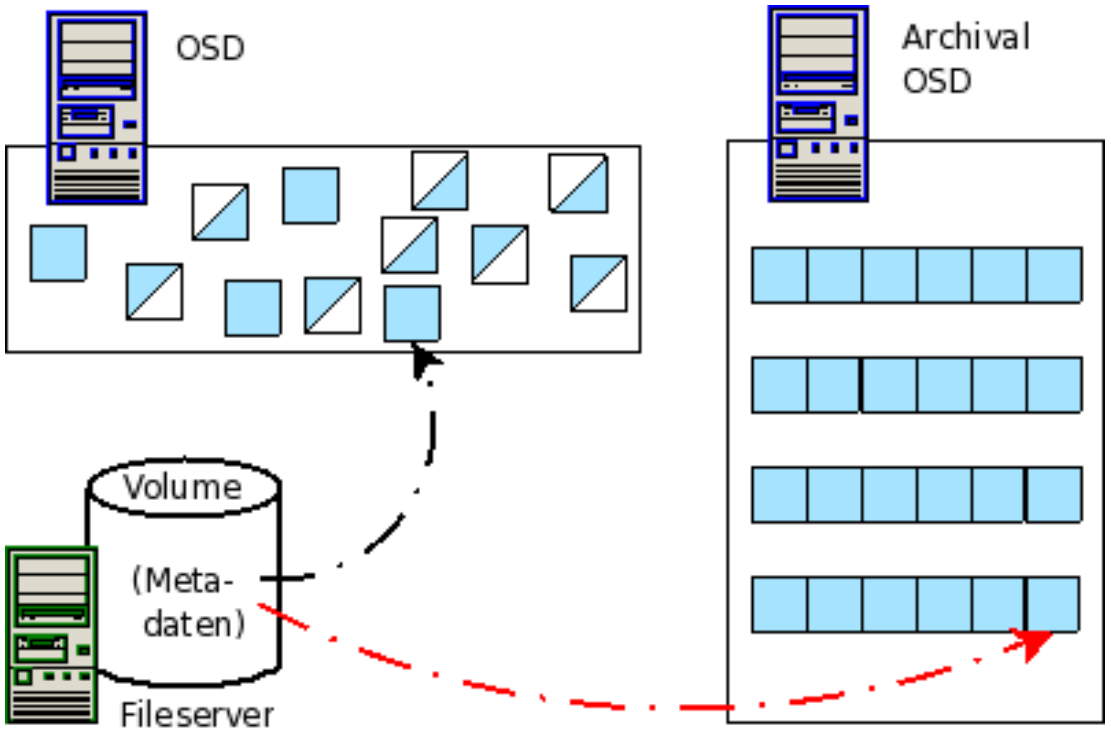
- OSD Dateien bestehen meist aus einem Objekt
- können aber auch gespiegelt oder verteilt sein
- z.B. zwei identische Objekte auf zwei OSDs (copies)
- z.B. zwei Hälften des Objekts (stripes)
- bis zu 8 Objekte pro Segment, z.B. 4 Stripes mit 2 Kopien



stripes=2,copies=2

Interface zu hierarchischem Storage

- spezieller Storage: “Archival OSD”
  - speichert Daten in z.B. TSM-HSM
- Files erhalten eine/mehrere Kopien in archival OSDs
- MD5-Summe ist in Metadaten auf AFS Fileserver
- Archivierung: `fs arch volume.c`
- aktive Kopie entfernen: `fs wipe volume.c`
  - bei Zugriff: transparentes Restore



Archivierung und Wiping

```
$ fs arch photos.tar
photos.tar done
$ fs osd photos.tar
photos.tar has 284 bytes of osd metadata, v=3
On-line, 1 segm, flags=0x0
  segment:
    lng=0, offs=0, stripes=1, strsize=0, cop=1, 1 objects
    object:
      obj=536870991.15108.1047363.0, osd=19, stripe=0
Archive, dv=1, 2009-03-02 13:48:02, 1 segm, flags=0x0
  segment:
    lng=2521335, offs=0, stripes=1, strsize=0, cop=1, 1 obj
    object:
      obj=536870991.15108.1047363.1, osd=18, stripe=0
  metadata:
    md5=45e2377a073b7e0b3367a1bcd4c0823d
      as from 2009-03-02 14:09:40
```


Status von OpenAFS+OSD

Stand der Entwicklung

- seit langem getestet
- in Garching seit über einem Jahr in Produktion
- bei DESY im Test, Produktion dieses Jahr

Beispielzelle: ipp-garching.mpg.de

- 46 AFS Fileserver, 30 OSDs
- ∼280 TB in AFS Space
- ∼550 TB in Object Storage
- Standorte in Garching und Greifswald
- Geographie wird berücksichtigt

Übersicht

Das AFS Dateisystem

- Vergleich mit NFSv3
- Grundlegende AFS-Konzepte
- Alternativ: Object Storage Systeme, Beispiel Lustre

Die Vereinigung: Das OpenAFS+OSD Projekt

- Idee und Funktion
- Nützliche Features von OpenAFS+OSD
- Fileserver Policies für OSD-Verwendung

Wofür Policies gebraucht werden

Größenkriterium zu schwach

- bekanntes Problem: `root` Tool
  - erzeugt grosse Dateien, schreibt jedoch zuerst nur 64-byte Header
  - Dateien sind aber an suffix `.root` erkennbar
- es wäre wünschenswert, Dateien am Namen zu identifizieren
- Zusatzfeature: automatisches Striping/Mirroring
  - ist ursprünglich nur mit umständlichem Kommando zugänglich
  - `fs createstripedfile libhello.a -stripes 2 -copies 2`
  - danach nutzt Schreiboperation die Stripes und Kopien
- Policies erkennen
  - Dateigröße, Dateiname (Wildcard- oder Regex), Benutzer und (AFS-)Gruppe
- Policies gelten
  - für ganzes Volume oder einzelnes Directory

DESY

Beispiele für Policies

Regeln: *Bedingung* $\Rightarrow$ *Ort, Stripes/Kopien*

- Ort: `osd, local, dynamic`
- Stripes/Kopien: 1,2,4,8 + Stripe-Size:12,13,...,19
- `stop` Option: wenn *Bedingung* dann beende Policy-Evaluierung

Beispiele

- $\sim$*"*.root"* $\Rightarrow$ `location=osd stop;`
- *true* $\Rightarrow$ `stripes=2;`
- $>10M \Rightarrow$ `copies=2;`
  $>100M \Rightarrow$ `copies=1,stripes=2;`
  } 2 Regeln

OSD Datenbank

- neuer Prozess auf Datenbankservern
- hält Policies und OSDs mit Prioritäten
- Policies können leicht ersetzt werden
- Fileserver hat immer Überblick über Platz auf OSDs

Referenzen

OpenAFS Download:

- *http://www.openafs.org/*

Mailing-Liste: openafs-info (at) openafs.org

- auch das DESY AFS-Team hilft gern per Mail

OpenAFS+OSD:

- *svn co https://svnsrv.desy.de/public/openafs-osd/trunk/*
- Pakete leider derzeit nur für Scientific Linux 5

Das OSD-Projekt wurde entwickelt von

- Hartmut Reuter (RZG)
- Rainer Többicke (CERN)
- Andrei Maslennikov (CASPUR)

Veranstaltungen:

- European AFS meeting 2009 in Rome (September)
  - *http://www.dia.uniroma3.it/∼afscon09/*
- ASF & Kerberos Best Practices Workshop (Juni)
  - *http://workshop.openafs.org/afsbpw09/cfp.html*

Danke für die Aufmerksamkeit

Zusätzliche AFS Dokumentation

- *http://www.openafs.org/doc/index.htm*
- Grand Central Org *http://grand.central.org/*
- AFS Wiki
  http://grand.central.org/twiki/bin/view/AFSLore/WebHome
- AFS-Seiten des CERN in
  http://consult.cern.ch/services/afs/
- Universitäten
  - *http://www.tu-braunschweig.de/it/services/sys/afs/*
  - *https://rz-static.uni-hohenheim.de/netzwerkbetriebssysteme/afs/kurs-folien/sahyoun/afs-einfuehrungskurs.ppt*
  - *http://www.tu-chemnitz.de/urz/kurse/unterlagen/admin-afs/basics.html*