

# Raid

# Agenda

1. Wer bin ich und warum ist Raid wichtig?
2. Allgemeine Definition
3. Raid Typen
4. Einsatzzweck
5. Implementierung
6. Fazit

Wer bin ich und warum ist Raid wichtig?

# Wer bin ich und warum ist Raid wichtig?

- Tests / Statistiken von über > 500.000 HDDs/SSDs erstellt
- Aufbau großer Setups mit viel Speicher (>2PB)
- Aufbau großer Setups mit vieeeeeeeeel IO Bedarf
- Erkenntnis 1: Datenträger gehen andauernd kaputt
- Erkenntnis 2: Datenträger können nicht schnell genug sein
- Zwangsweise mit Raid beschäftigt

# Allgemeine Definition

# Allgemeine Definition

- Redundant Array of Independent Disks
- Redundant Array of Inexpensive Disks
- Logischer Zusammenschluss für mehr Leistung und Ausfallsicherheit
- Daten werden in Chunks/Stripes aufgeteilt
- Standard Chunk size von 64kB
- Größere Dateien => größere Chunks

# Allgemeine Definition

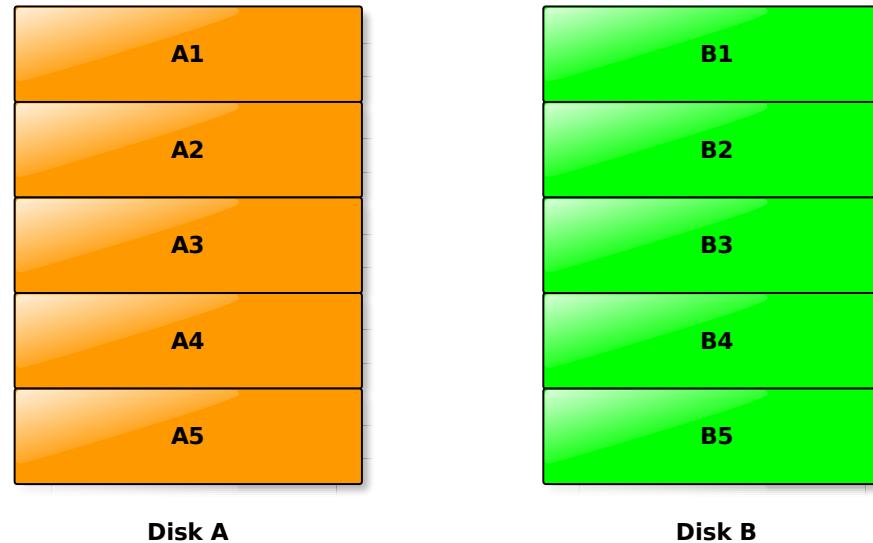
## - Striping

- Nutzbarer Speicher [Kapazität der kleinsten Disk] \* N
- $N > 1$
- Chunks werden gleichmäßig auf alle Datenträger verteilt
- Schreiben/Lesen kann parallel auf allen Datenträgern erfolgen
- Alle Daten verloren beim Ausfall eines Datenträgers
- Eigentlich gar kein Raid weil !Redundant

# Allgemeine Definition

- Striping

Raid 0



Disk A

Disk B

# Allgemeine Definition

- Striping

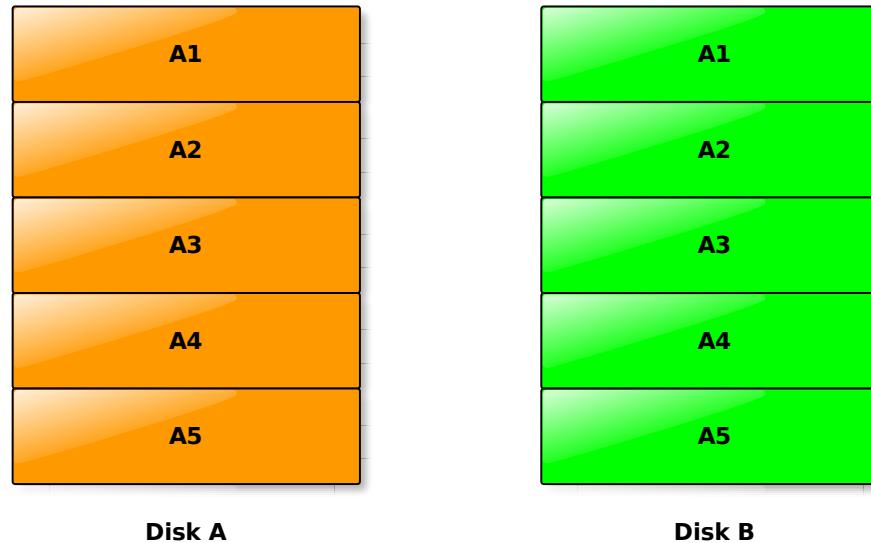
- Nutzbarer Speicher [Kapazität der kleinsten Disk]
- $N > 1$
- Ein Chunk wird auf allen Disks gespeichert
- Lesen kann parallel auf allen Datenträgern erfolgen
- Das ist Raid :)

- Mirroring

# Allgemeine Definition

- Striping
- Mirroring

Raid 1



# Allgemeine Definition

- Striping

- Nutzbarer Speicher [Kapazität der kleinsten Disk] \* (N-1)
- $N > 2$
- Parallel Schreiben von N-1 Chunks (Paritychunk wird danach geschrieben)
- Paritychunk = XOR(N-1 Chunks)

- Mirroring

- Lesen/Schreiben parallel auf allen Datenträgern
- Das ist auch noch Raid :)

- Parity

# Allgemeine Definition

- Striping

- Mirroring

- Parity

Update eines Chunks:

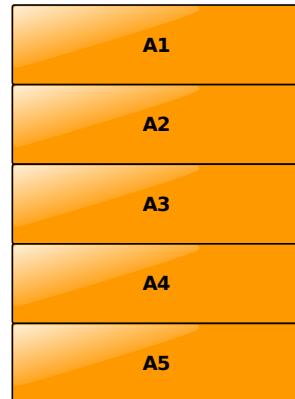
- Lesen des Chunks
- Vergleich ob neuer Chunk != alter Chunk
- Lesen des Parity Chunks
- neuer Paritychunk = XOR(alter Chunk + neuer Chunk + Parity Chunk)
- Schreiben vom neuen Chunk + neuer Paritychunk

# Allgemeine Definition

- Striping

- Mirroring

- Parity

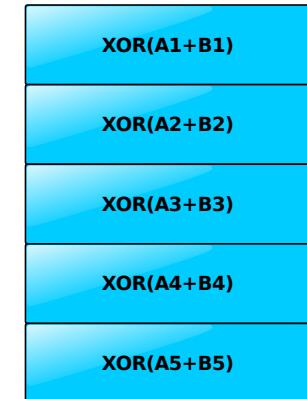


Disk A

## Raid 4



Disk B



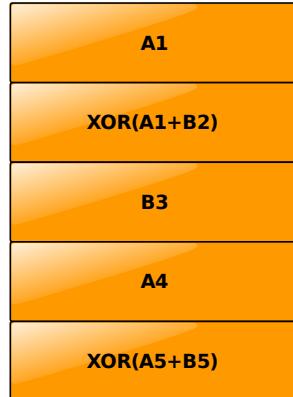
Disk C

# Allgemeine Definition

- Striping

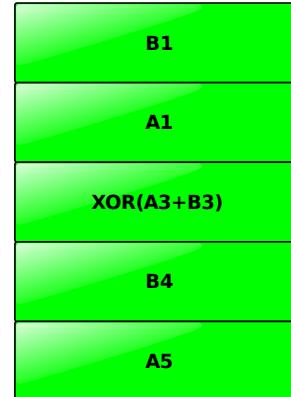
- Mirroring

- Parity



Disk A

## Raid 5



Disk B



Disk C

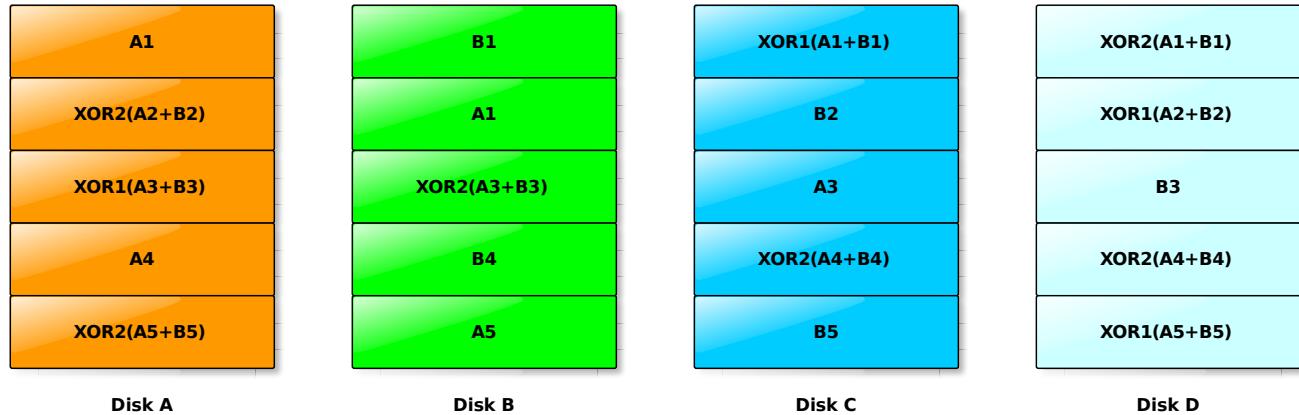
# Allgemeine Definition

- Striping

- Mirroring

- Parity

Raid 6



# Einsatzzweck

## Einsatzzweck

## Raidlevel

Vorteile:

- Hohe Schreib/Leseleistung

Nachteile:

### - Raid 0

- Kein Schutz vor Hardwaredefekten

Einsatzzweck:

- Speichern von OS/Spiele im Heimbereich

# Einsatzzweck

## Raidlevel

- Raid 0

- Raid 1

Vorteile:

- Hohe Leseleistung
- N-1 Disks können ausfallen ohne Datenverlust

Nachteile:

- Große Differenz zwischen Netto/Brutto Kapazität

Einsatzzweck:

- OS Installation im Server/Workstation Bereich
- Datenspeicher in Servern mit geringen Schreibanforderungen

# Einsatzzweck

## Raidlevel

- Raid 0

- Raid 1

- Raid 4

Vorteile:

- erhöhte Schreib/Leseleistung
- Eine Disk kann ausfallen ohne Datenverlust

Nachteile:

- Parity Berechnung kostet Rechenleistung
- Ausfall Parity Disk => Neuberechnung Parity Chunks
- Starke Belastung aller verbliebenen Disks bei Rebuild

Einsatzzweck:

- Hohe Belastung beim Ausfall der Parity Disk => Garnicht

# Einsatzzweck

## Raidlevel

- Raid 0

- Raid 1

- Raid 4

- Raid 5

Vorteile:

- erhöhte Schreib/Leseleistung
- Eine Disk kann ausfallen ohne Datenverlust

Nachteile:

- Parity Berechnung kostet Rechenleistung
- Hohe Belastung aller verbliebenen Disks bei Rebuild

Einsatzzweck:

- Große Datenspeicher bis ~10TB mit geringen Random IO Anforderungen

# Einsatzzweck

## Raidlevel

- Raid 0

- Raid 1

- Raid 4

- Raid 5

- Raid 6

Vorteile:

- erhöhte Schreib/Leseleistung
- Zwei Disks können ausfallen ohne Datenverlust

Nachteile:

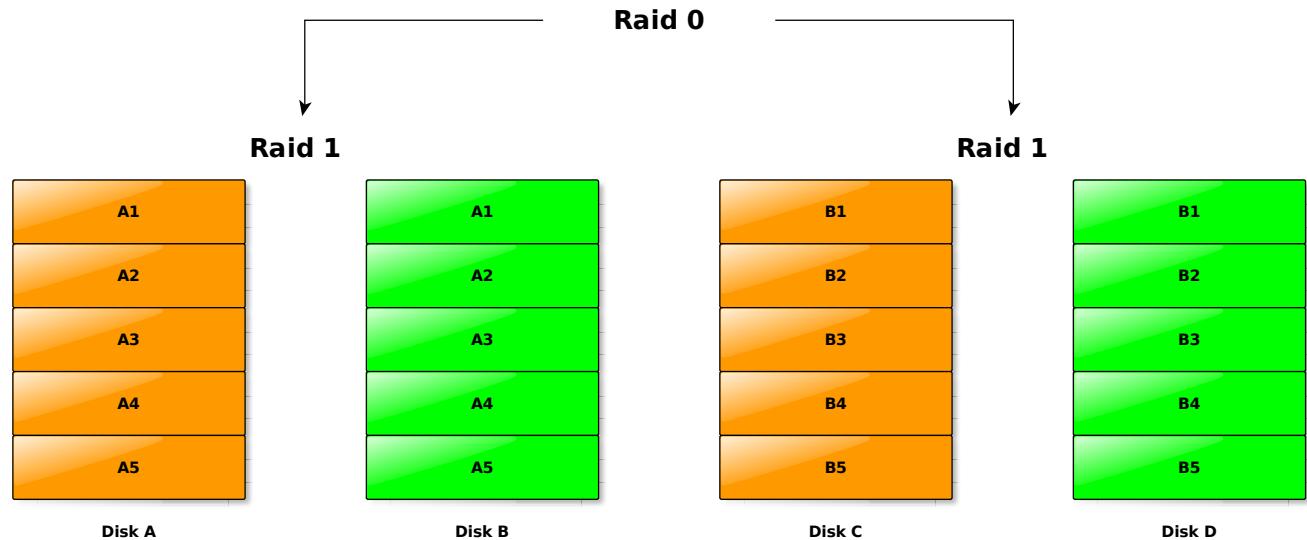
- Parity Berechnung kostet Rechenleistung
- Hohe Belastung aller verbliebenen Disks bei Rebuild

Einsatzzweck:

- Große Datenspeicher bis ~19TB mit geringen Random IO Anforderungen

# Einsatzzweck Raidkombis

- Raid 10



## Einsatzzweck

### Raidkombis

#### - Raid 10

Vorteile:

- Hohe Schreib/Leseleistung
- Gute Skalierbarkeit
- Ausfall mehrerer Disks möglich
- Geringe Rebuildbelastung
- Sehr gute Random IO Leistung

Nachteile:

- Hoher Brutto/Netto Unterschied
- Beim Rebuild wird die einzige Datenplatte belastet

Einsatzzweck:

- Virtualisierung
- Datenbanken
- Alles mit viel Random IO

## Einsatzzweck

## Raidkombis

- Raid 10

- Rest

- Raid 01
- Raid 50
- Raid 51
- Raid 60
- Raid 61

Hardware (RoC/Fakeraid)

Software (mdadm/Dateisysteme)

# Hardware

- RoC



# Hardware

- RoC



# Hardware

- RoC

- Fakeraid

- Raid Algorithmen gespeichert im Chipsatz(BiosRaid)
- Raid Algorithmen gespeichert auf Zusatzkarte(FakeRaid)
- Berechnungen übernimmt CPU, kein dedizierter Controller

# Software

## - mdadm

- Raid Algorithmen im Linux Kernel
- mdadm = **M**ultiple **D**isk **A**DMINistration
- mdadm => Userlandtool zum ansprechen der Algorithmen
- Verwaltung von N Blockdevices in einem Raidverbund

# Software

## - mdadm

Vorteile:

- Freie Implementierung der Algorithmen
- Raids sind unabhängig von Controller/CPU

Nachteile:

- Geringere Performance zu RoC bei Raidkombinationen (nur im HPC Bereich)

Einsatzzweck:

- Jeder Linux Server/Workstation (außer HPC)

# Software

## - mdadm

```
[root@bastelfreak-fs ~]# cat /proc/mdstat
Personalities : [raid1] [raid6] [raid5] [raid4]
md125 : active raid6 sda[4] sdd[7] sdi[8] sdg[2] sdf[3]
        sdh[1] sde[0] sdc[6] sdb[5]
        20510945280 blocks super 1.2 level 6, 512k chunk,
        algorithm 2 [9/9] [UUUUUUUUUU]
        bitmap: 0/22 pages [0KB], 65536KB chunk

md126 : active raid1 sdj2[0] sdm2[3] sdl2[2] sdk2[1]
        1049536 blocks super 1.0 [4/4] [UUUU]
        bitmap: 1/1 pages [4KB], 65536KB chunk

md127 : active raid1 sdj1[0] sdm1[3] sdl1[2] sdk1[1]
        52510592 blocks super 1.2 [4/4] [UUUU]
        bitmap: 1/1 pages [4KB], 65536KB chunk

unused devices: <none>
[root@bastelfreak-fs ~]#
```

# Software

Was ist das?

- mdadm

- Zettabyte File System von Sun
- Dateisystem inkl Volume Manager und Raid Support

- ZFS

- Konzipiert für den Enterprise Einsatz (Backupstorage mit >2PB, riesige Virtualisierungscluster)
- Funktioniert aber auch toll Zuhause :)
- Entwickelt auf FreeBSD und Solaris
- Portierung auf Linux

# Software

Was kann ZFS als Dateisystem?

- mdadm

- Maximale Dateigröße von 16EiB ( $2^{60}$ )

- ZFS

- Maximale Dateisystemgröße 16EiB
- Maximal  $2^{48}$  Dateien
- Transparente Komprimierung
- Transparente Verschlüsselung
- Dynamische Inodes
- Checksumming (CRC32)
- Scrubbing
- Deduplication
- Snapshots (CoW)

# Software

Was kann ZFS als Volume Manager?

- Partial Rebuild
- Send/Receive
- Pools mit dynamischen Partitionsgrößen
- Kein Write Hole bei Raid-Z1, Raid-Z2, Raid-Z3
- ZIL - ZFS Intent Log Cache (Ram oder Blockdevice[s])
- ARC - Adaptive Replacement Cache (Ram)
- L2ARC - Layer 2 Adaptive Replacement Cache (Ram oder Blockdevice[s])

# Software

Wie ZFS aber:

- mdadm

- Native Linux Entwicklung
- Keine eingebaute Verschlüsselung

- ZFS

- Online Raid Level Migration
- Unterschiedliche Raidlevel für Daten und Metadaten

- BTRFS

# Software

- mdadm

- ZFS

- BTRFS

- glusterfs

Netzwerkdateisystem:

- Network Storage
- Distributed Storage
- Replicated Storage
- Distributed Replicated Storage

# Software

## Distributed Object Store:

- Speicherung von Binärobjekten
  - Ceph-Layer auf normalen Datenträgern (OSD - Object Storage Daemon)
- mdadm
- MONs verwalten Crushmap
  - MONs übernehmen Fencing und Quorum (ohne STONITH), verhindern Split-Brain
- ZFS
- BTRFS
- glusterfs
- ceph

# Software

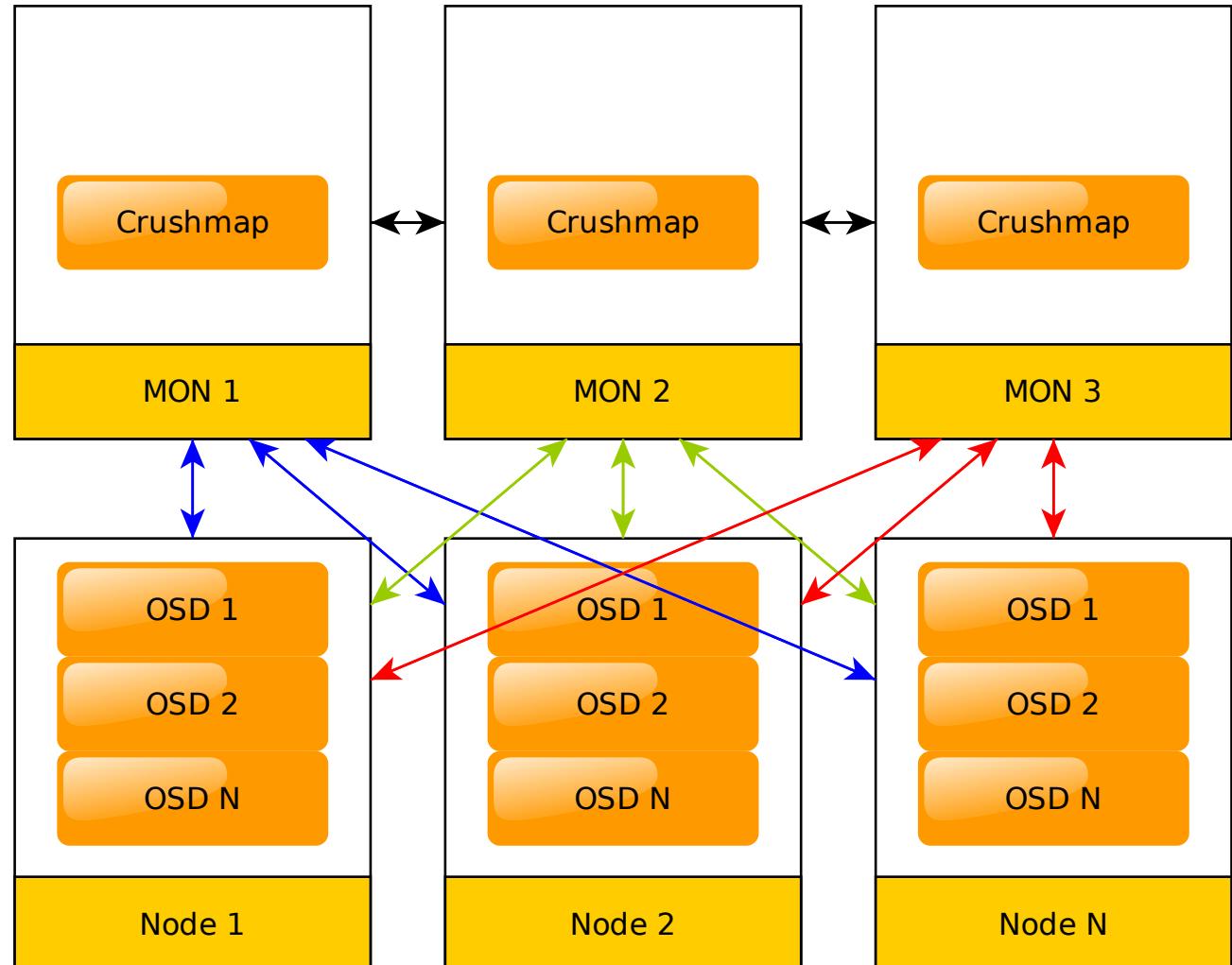
- mdadm

- ZFS

- BTRFS

- glusterfs

- ceph



# Fazit

Raid bietet mehr logischen Speicher, Ausfallsicherheit, gesteigerte IO Performance

Raid ersetzt keine Backups!