DAAD Summerschool Curitiba 2011

Aspects of Large Scale High Speed Computing Building Blocks of a Cloud

Storage Networks

2: Virtualization of Storage: RAID, SAN and Virtualization

Christian Schindelhauer

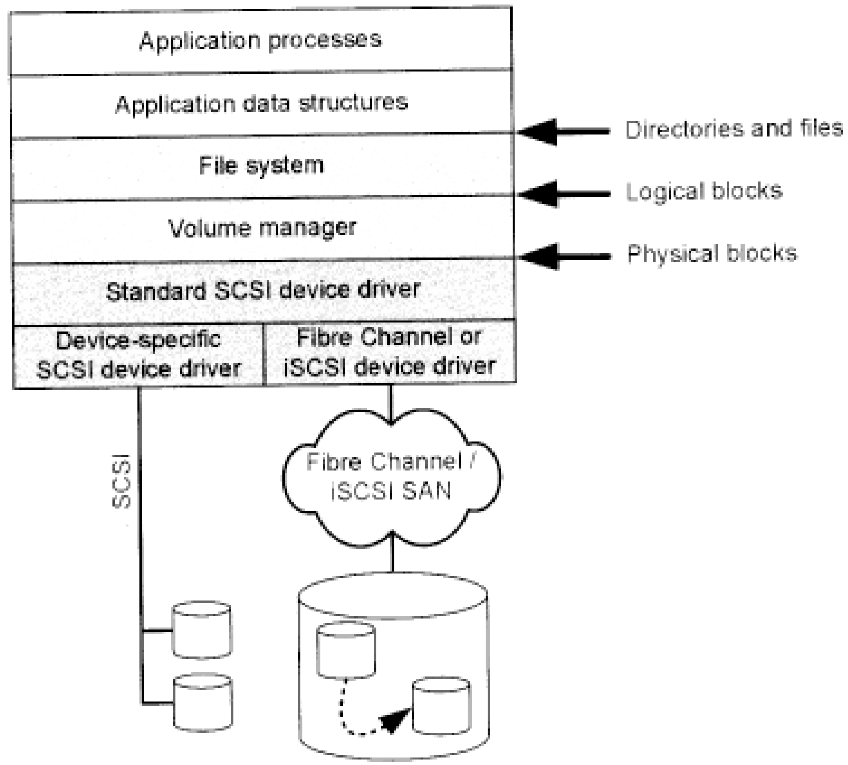
Technical Faculty

Computer-Networks and Telematics

University of Freiburg

Volume Manager

- **Volume manager**
  - aggregates physical hard disks into virtual hard disks
  - breaks down hard disks into smaller hard disks
  - Does not provide files system, but enables it
- **Can provide**
  - resizing of volume groups by adding new physical volumes
  - resizing of logical volumes
  - snapshots
  - mirroring or striping, e.g. like RAID1
  - movement of logical volumes



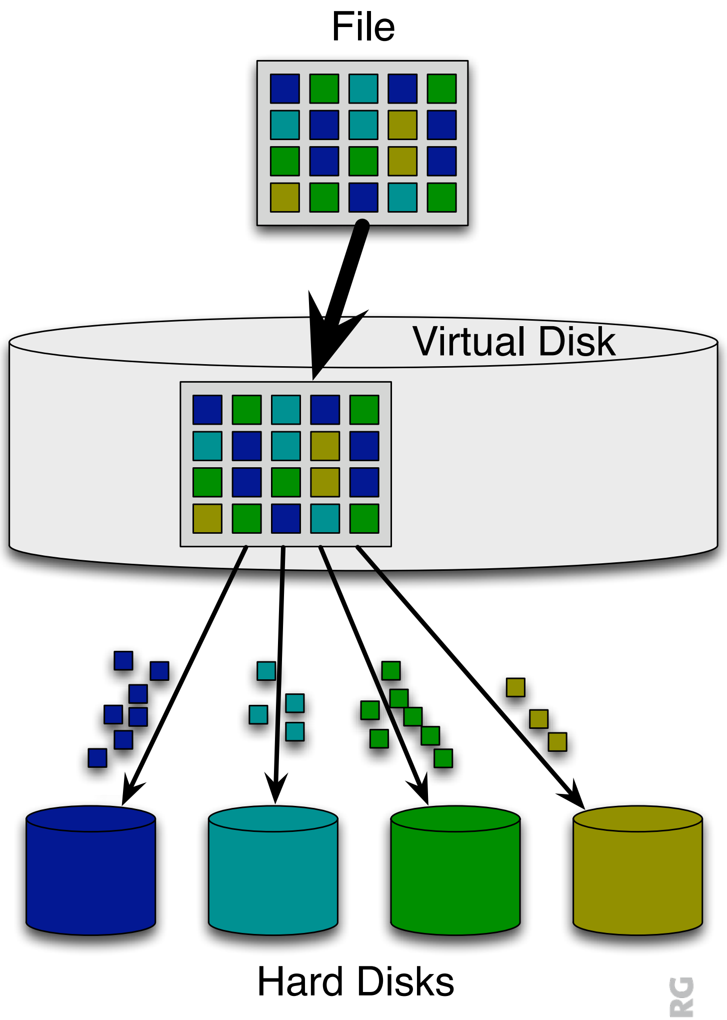
From: Storage Networks Explained, Basics and Application of Fibre Channel SAN, NAS, iSCSI and InfiniBand, Troppens, Erkens, Müller, Wiley

Overview of Terms

- Physical volume (PV)
  - hard disks, RAID devices, SAN
- Physical extents (PE)
  - Some volume managers splite PVs into same-sized physical extents
- Logical extent (LE)
  - physical extents may have copies of the same information
  - are addresed as logical extent
- Volume group (VG)
  - logical extents are grouped together into a volume group
- Logical volume (LV)
  - are a concatenation of volume groups
  - a raw block devices
  - where a file system can be created upon

Concept of Virtualization

- **Principle**
  - A virtual storage constitutes handles all application accesses to the file system
  - The virtual disk partitions files and stores blocks over several (physical) hard disks
  - Control mechanisms allow redundancy and failure repair
- **Control**
  - Virtualization server assigns data, e.g. blocks of files to hard disks (address space remapping)
  - Controls replication and redundancy strategy
  - Adds and removes storage devices



Storage Virtualization

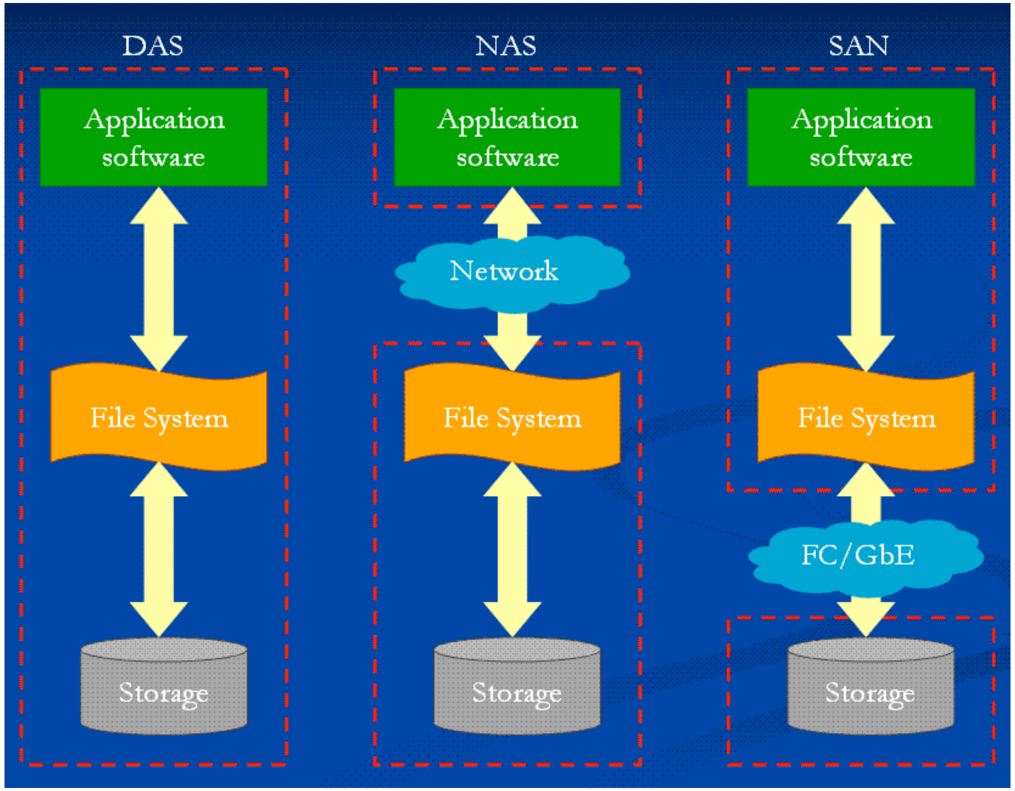
- Capabilities
  - Replication
  - Pooling
  - Disk Management
- Advantages
  - Data migration
  - Higher availability
  - Simple maintenance
  - Scalability
- Disadvantages
  - Un-installing is time consuming
  - Compatibility and interoperability
  - Complexity of the system
- Classic Implementation
  - Host-based
    - Logical Volume Management
    - File Systems, e.g. NFS
  - Storage devices based
    - RAID
  - Network based
    - Storage Area Network
- New approaches
  - Distributed Wide Area Storage Networks
  - Distributed Hash Tables
  - Peer-to-Peer Storage

Storage Area Networks

- Virtual Block Devices
  - without file system
  - connects hard disks
- Advantages
  - simpler storage administration
  - more flexible
  - servers can boot from the SAN
  - effective disaster recovery
  - allows storage replication
- Compatibility problems
  - between hard disks and virtualization server

SAN Networking

- **Networking**
  - FCP (Fibre Channel Protocol)
    - SCSI over Fibre Channel
  - iSCSI (SCSI over TCP/IP)
  - HyperSCSI (SCSI over Ethernet)
  - ATA over Ethernet
  - Fibre Channel over Ethernet
  - iSCSI over InfiniBand
  - FCP over IP



http://en.wikipedia.org/wiki/Storage_area_network

SAN File Systems

- File system for concurrent read and write operations by multiple computers
  - without conventional file locking
  - concurrent direct access to blocks by servers
- Examples
  - Veritas Cluster File System
  - Xsan
  - Global File System
  - Oracle Cluster File System
  - VMware VMFS
  - IBM General Parallel File System

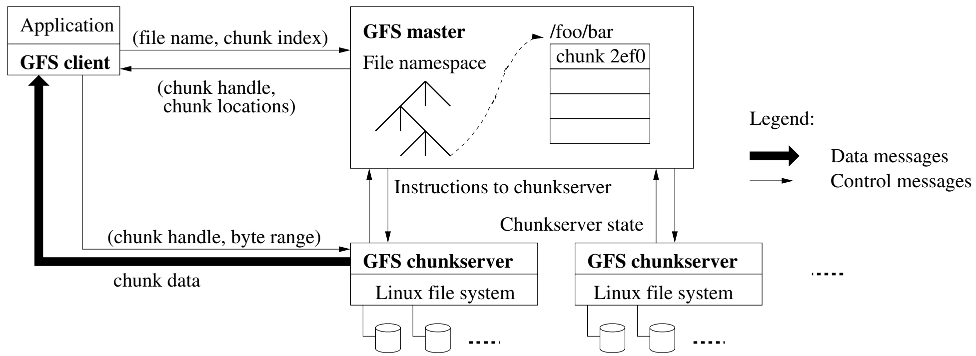
Distributed File Systems (without Virtualization)

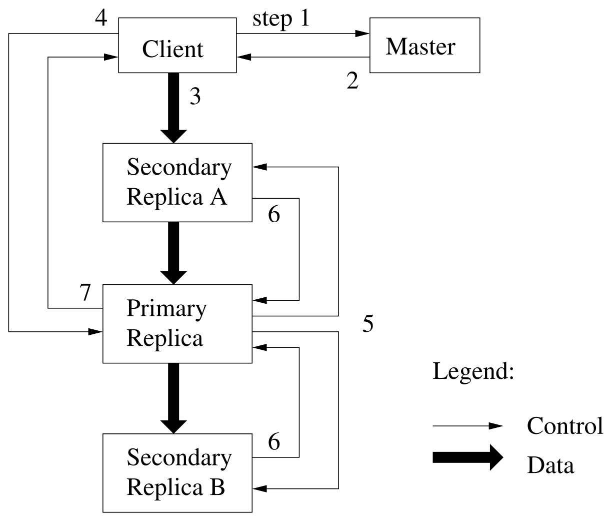
- aka. Network File System
- Supports sharing of files, tapes, printers etc.
- Allows multiple client processes on multiple hosts to read and write the same files
  - concurrency control or locking mechanisms necessary
- Examples
  - Network File System (NFS)
  - Server Message Block (SMB), Samba
  - Apple Filing Protocol (AFP)
  - Amazon Simple Storage Service (S3)

Distributed File Systems with Virtualization

CoNe Freiburg

- **Example: Google File System**
- **File system on top of other file systems with builtin virtualization**
  - System built from cheap standard components (with high failure rates)
  - Few large files
  - Only operations: read, create, append, delete
    - concurrent appends and reads must be handled
  - High bandwidth important
- **Replication strategy**
  - chunk replication
  - master replication





The Google File System
Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung

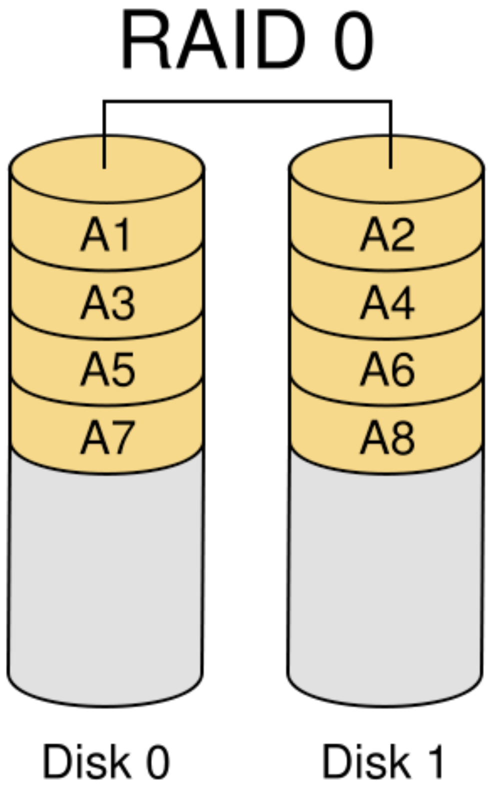
UNI FREIBURG

RAID

- Redundant Array of Independent Disks
  - Patterson, Gibson, Katz, „A Case for Redundant Array of Inexpensive Disks“, 1987
- Motivation
  - Redundancy
    - error correction and fault tolerance
  - Performance (transfer rates)
  - Large logical volumes
  - Exchange of hard disks, increase of storage during operation
  - Cost reduction by use of inexpensive hard disks

Raid 0

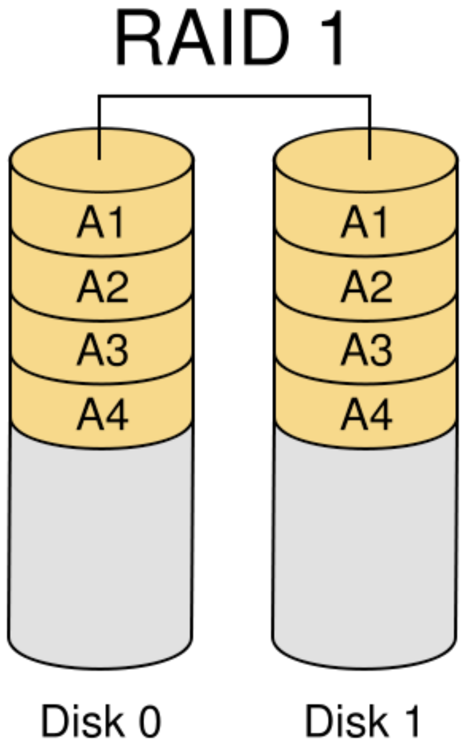
- **Striped set without parity**
  - Data is broken into fragments
  - Fragments are distributed to the disks
- **Improves transfer rates**
- **No error correction or redundancy**
- **Greater disk of data loss**
  - compared to one disk
- **Capacity fully available**



http://en.wikipedia.org/wiki/RAID

Raid 1

- **Mirrored set without parity**
  - Fragments are stored on all disks
- **Performance**
  - if multi-threaded operating system allows split seeks then
  - faster read performance
  - write performance slightly reduced
- **Error correction or redundancy**
  - all but one hard disks can fail without any data damage
- **Capacity reduced by factor 2**



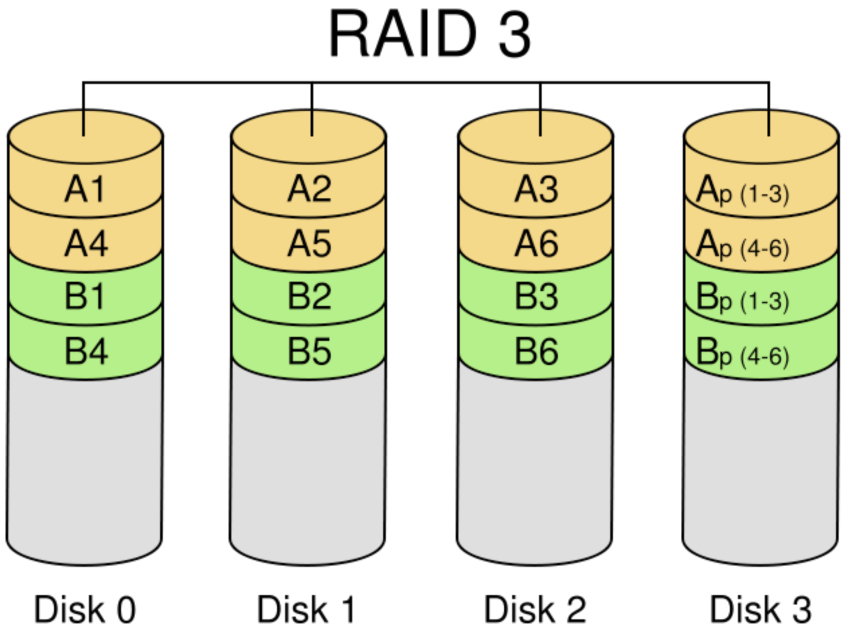
http://en.wikipedia.org/wiki/RAID

RAID 2

- Hamming Code Parity
- Disks are synchronized and striped in very small stripes
- Hamming codes error correction is calculated across corresponding bits on disks and stored on multiple parity disks
- not in use

Raid 3

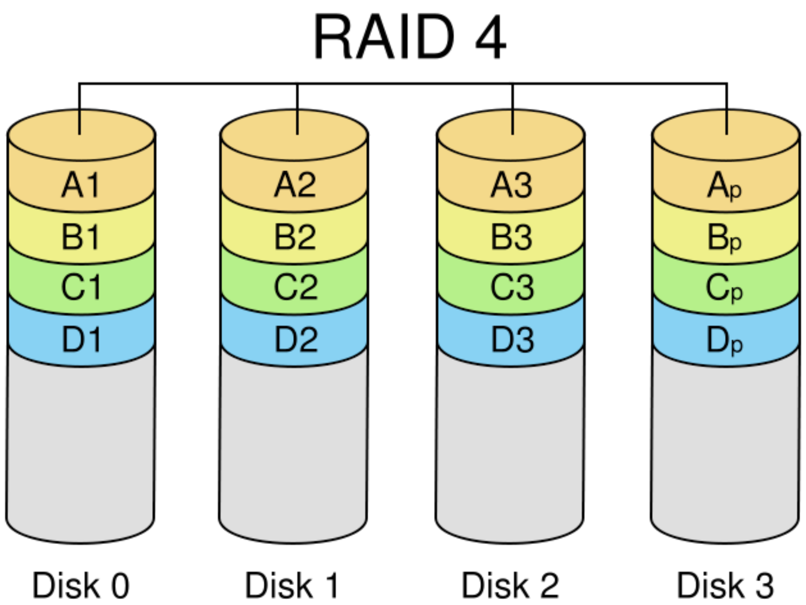
- **Striped set with dedicated parity (byte level parity)**
  - Fragments are distributed on all but one disks
  - One dedicated disk stores a parity of corresponding fragments of the other disks
- **Performance**
  - improved read performance
  - write performance reduced by bottleneck parity disk
- **Error correction or redundancy**
  - one hard disks can fail without any data damage
- **Capacity reduced by 1/n**



http://en.wikipedia.org/wiki/RAID

Raid 4

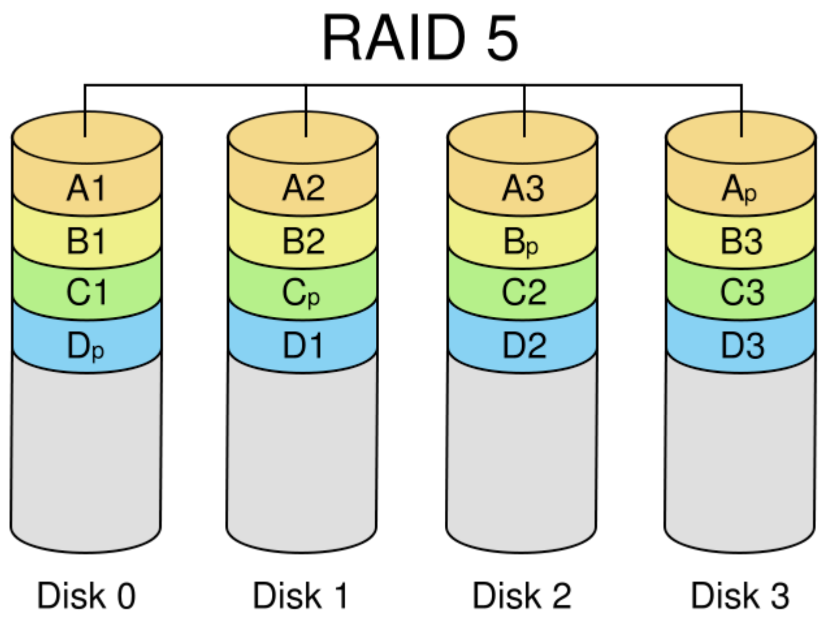
- **Striped set with dedicated parity (block level parity)**
  - Fragments are distributed on all but one disks
  - One dedicated disk stores a parity of corresponding blocks of the other disks on I/O level
- **Performance**
  - improved read performance
  - write performance reduced by bottleneck parity disk
- **Error correction or redundancy**
  - one hard disks can fail without any data damage
- **Hardly in use**



http://en.wikipedia.org/wiki/RAID

Raid 5

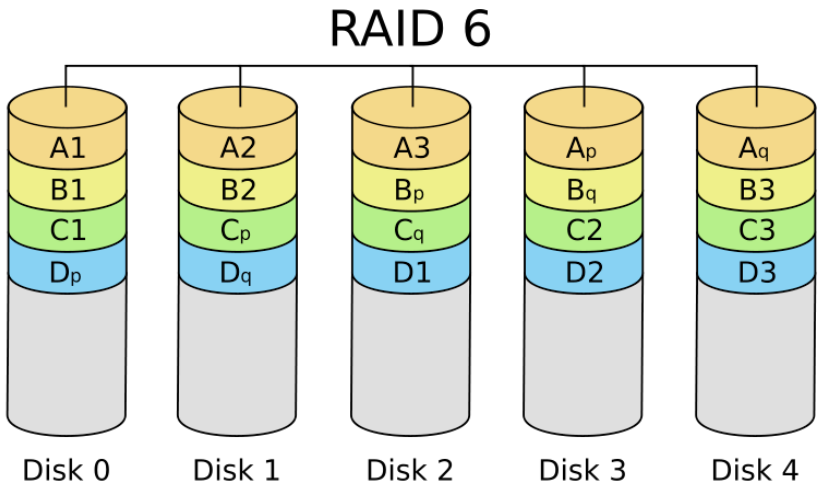
- **Striped set with distributed parity (interleave parity)**
  - Fragments are distributed on all but one disks
  - Parity blocks are distributed over all disks
- **Performance**
  - improved read performance
  - improved write performance
- **Error correction or redundancy**
  - one hard disks can fail without any data damage
- **Capacity reduced by 1/n**



http://en.wikipedia.org/wiki/RAID

Raid 6

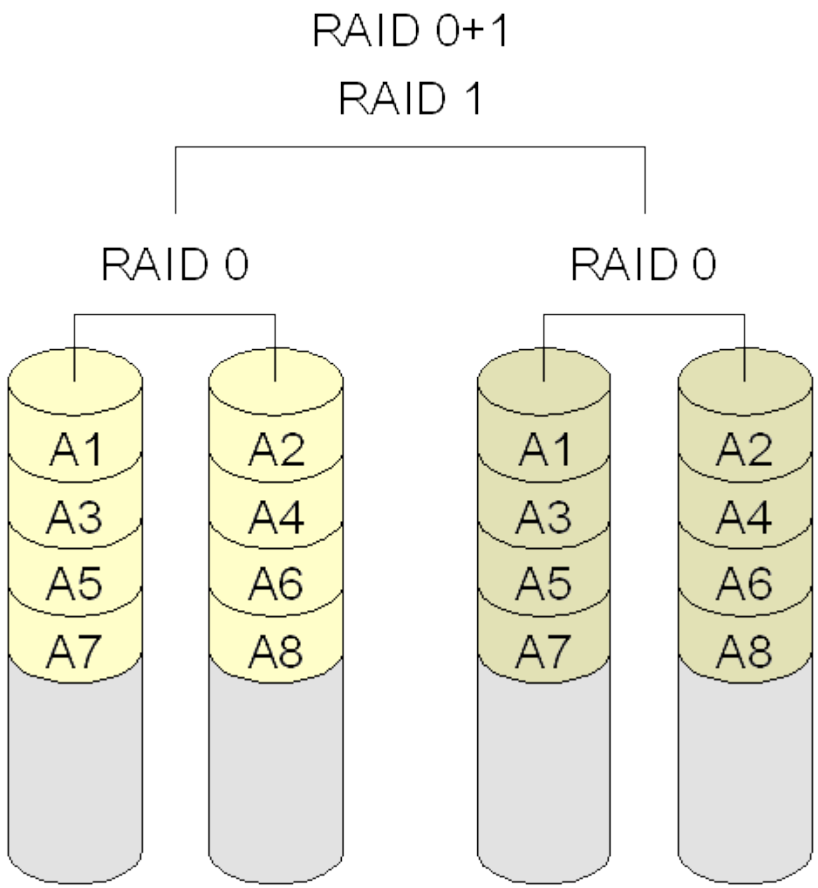
- **Striped set with dual distributed parity**
  - Fragments are distributed on all but two disks
  - Parity blocks are distributed over two of the disks
    - one uses XOR other alternative method
- **Performance**
  - improved read performance
  - improved write performance
- **Error correction or redundancy**
  - two hard disks can fail without any data damage
- **Capacity reduced by 2/n**



http://en.wikipedia.org/wiki/RAID

RAID 0+1

- **Combination of RAID 1 over multiple RAID 0**
- **Performance**
  - improved because of parallel write and read
- **Redundancy**
  - can deal with any single hard disk failure
  - can deal up to two hard disk failure
- **Capacity reduced by factor 2**

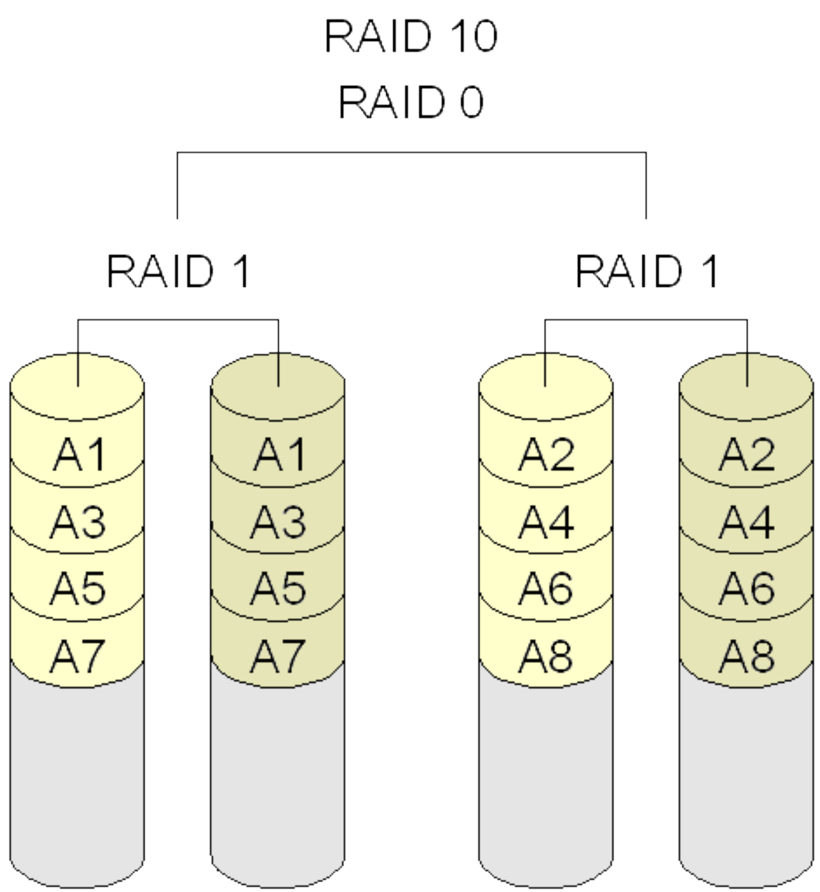


http://en.wikipedia.org/wiki/RAID

CoNe Freiburg

RAID 10

- **Combination of RAID 0 over multiple RAID 1**
- **Performance**
  - improved because of parallel write and read
- **Redundancy**
  - can deal with any single hard disk failure
  - can deal up to two hard disk failure
- **Capacity reduced by factor 2**



http://en.wikipedia.org/wiki/RAID

More RAIDs

- More:
  - RAIDn, RAID 00, RAID 03, RAID 05, RAID 1.5, RAID 55, RAID-Z, ...
- Hot Swapping
  - allows exchange of hard disks during operation
- Hot Spare Disk
  - unused reserve disk which can be activated if a hard disk fails
- Drive Clone
  - Preparation of a hard disk for future exchange indicated by S.M.A.R.T

RAID
Waterproof Definitions


Standalone


Cluster


Hot swap


RAID 0


RAID 1


RAID 5

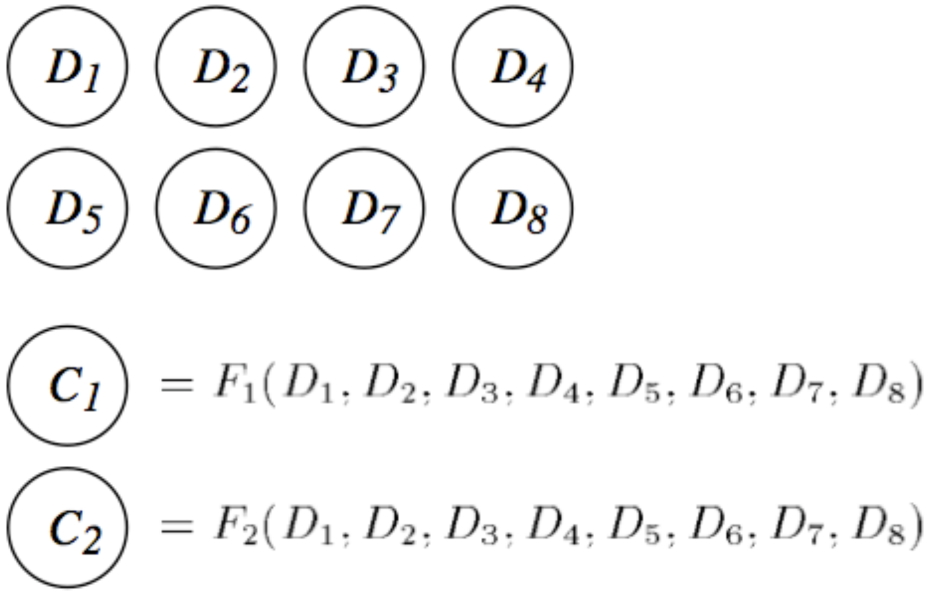

RAID 0+1

Raid-6 Encodings

- A Tutorial on Reed-Solomon Coding for Fault-Tolerance in RAID-like Systems, James S. Plank , 1999
- The RAID-6 Liberation Codes, James S. Plank, FAST´08, 2008

Principle of RAID 6

- **Data units $D_1, \ldots, D_n$**
  - w: size of words
    - w=1 bits,
    - w=8 bytes, ...
- **Checksum devices $C_1, C_2, \ldots, C_m$**
  - computed by functions $C_i = F_i(D_1, \ldots, D_n)$
- **Any n words from data words and check words**
  - can decode all n data units



A Tutorial on Reed-Solomon Coding for Fault-Tolerance in RAID-like Systems, James S. Plank , 1999

Principle of RAID 6

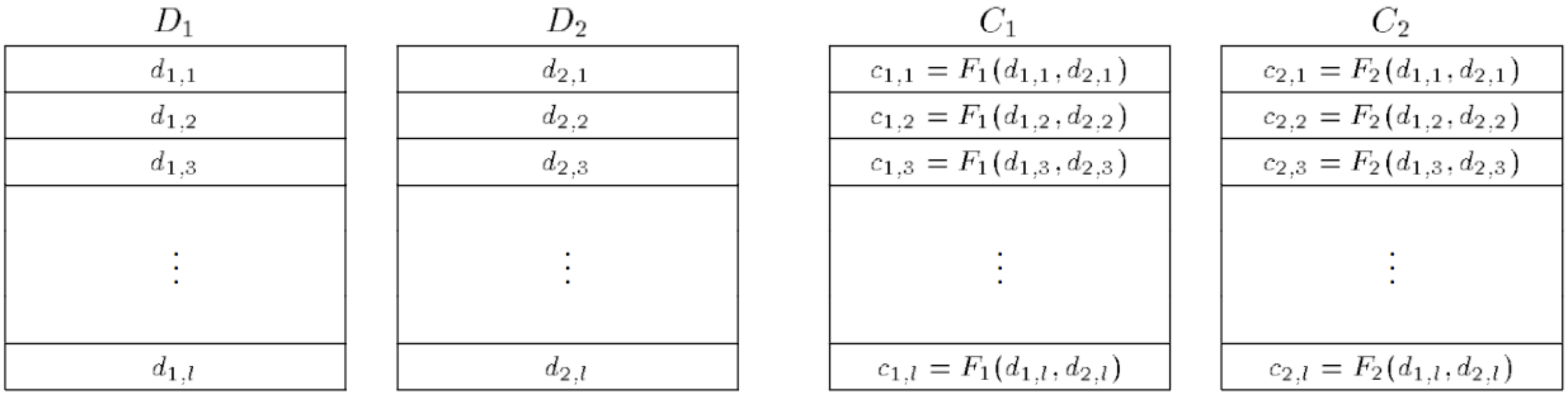


Figure 2: Breaking the storage devices into words ($n = 2, m = 2, l = \frac{8k}{w}$)

A Tutorial on Reed-Solomon Coding for Fault-Tolerance
in RAID-like Systems, James S. Plank , 1999

Operations

- Encoding
  - Given new data elements, calculate the check sums
- Modification (update penalty)
  - Recompute the checksums (relevant parts) if one data element is modified
- Decoding
  - Recalculate lost data after one or two failures
- Efficiency
  - speed of operations
  - check disk overhead
  - ease of implementation and transparency

Reed-Solomon

- RAID 6 Encodings

Vandermonde-Matrix

$$\begin{bmatrix} f_{1,1} & f_{1,2} & \dots & f_{1,n} \\ f_{2,1} & f_{2,2} & \dots & f_{2,n} \\ \vdots & \vdots & & \vdots \\ f_{m,1} & f_{m,2} & \dots & f_{m,n} \end{bmatrix} \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_n \end{bmatrix} =$$

$$\begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & 2 & 3 & \dots & n \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & 2^{m-1} & 3^{m-1} & \dots & n^{m-1} \end{bmatrix} \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_n \end{bmatrix} = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_m \end{bmatrix}$$

A Tutorial on Reed-Solomon Coding for Fault-Tolerance in RAID-like Systems, James S. Plank , 1999

Complete Matrix

$$\begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \dots & 1 \\ 1 & 1 & 1 & \dots & 1 \\ 1 & 2 & 3 & \dots & n \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & 2^{m-1} & 3^{m-1} & \dots & n^{m-1} \end{bmatrix} \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_n \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_n \\ c_1 \\ c_2 \\ \vdots \\ c_m \end{bmatrix}$$

A Tutorial on Reed-Solomon Coding for Fault-Tolerance in RAID-like Systems, James S. Plank , 1999

Galois Fields

- $GF(2^w)$ = Finite Field over 2^w elements
  - Elements are all binary strings of length w
  - $0 = 0^w$ is the neutral element for addition
  - $1 = 0^{w-1}1$ is the neutral element for multiplication
- u + v = bit-wise Xor of the elements
  - e.g. 0101 + 1100 = 1001
- a b= product of polynomials modulo 2 and modulo an irreducible polynomial q
  - i.e. $(a_{w-1} \ldots a_1\, a_0)\,(b_{w-1} \ldots b_1\, b_0) =$

$$((a_0 + a_1 x + \ldots + a_{w-1} x^{w-1})(b_0 + b_1 x + \ldots + b_{w-1} x^{w-1}) \bmod q(x)) \bmod 2)$$

Example: $GF(2^2)$

CoNe Freiburg

Generated Element of $GF(4)$	Polynomial Element of $GF(4)$	Binary Element b of $GF(4)$	Decimal Representation of b
0	0	00	0
x^0	1	01	1
x^1	x	10	2
x^2	$x+1$	11	3

+	0 = 00	1 = 01	2 = 10	3 = 11
0 =00	0	1	2	3
1 =01	1	0	3	2
2 =10	2	3	0	1
3 =11	3	2	1	0

$q(x) = x^2+x+1$

*	0 = 0	1 = 1	2 = x	3 = x+1
0 = 0	0	0	0	0
1 = 1	0	1	2	3
2 = x	0	2	3	1
3 = x+1	0	3	1	2

$2 \cdot 3 = x(x+1) = x^2+x = 1 \bmod x^2+x+1 = 1$

$2 \cdot 2 = x^2 = x+1 \bmod x^2+x+1 = 3$

UNI FREIBURG

Irreducible Polynomials

- Irreducible polynomials cannot be factorized
  - counter-example: $x^2+1 = (x+1)^2 \bmod 2$
- Examples:
  - w=2: x^2+x+1
  - w=4: x^4+x+1
  - w=8: $x^8+x^4+x^3+x^2+1$
  - w=16: $x^{16}+x^{12}+x^3+x+1$
  - w=32: $x^{32}+x^{22}+x^2+x+1$
  - w=64: $x^{64}+x^4+x^3+x+1$

Fast Multiplication

- Powers laws
  - Consider: $\{2^0, 2^1, 2^2, \ldots\}$
  - $= \{x^0, x^1, x^2, x^3, \ldots$
  - $=$ exp(0), exp(1), ...
- exp(x+y) = exp(x) exp(y)
- Inverse: log(exp(x)) = x
  - $\log(x \cdot y) = \log(x) + \log(y)$
- x y = exp(log(x) + log(y))
  - Warning: integer addition!!!
- Use tables to compute exponential and logarithm function

Example: GF(16)

$q(x) = x^4+x+1$

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
exp(x)	1	x	x^2	x^3	$1+x$	$x+x^2$	x^2+x^3	$1+x+x^3$	$1+x^2$	$x+x^3$	$1+x+x^2$	$x+x^2+x^3$	$1+x+x^2+x^3$	$1+x^2+x^3$	$1+x^3$	1
exp(x)	1	2	4	8	3	6	12	11	5	10	7	14	15	13	9	1

x	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
log(x)	0	1	4	2	8	5	10	3	14	9	7	6	13	11	12

- $5 \cdot 12$ = exp(log(5)+log(12)) = exp(8+6) = exp(14) = 9
- $7 \cdot 9$ = exp(log(7)+log(9)) = exp(10+14) = exp(24) = exp(24-15)
  = exp(9) = 10

Example: Reed Solomon for GF[2^4]

- Compute carry bits for three hard disks by computing

$$F = \begin{bmatrix} 1^0 & 2^0 & 3^0 \\ 1^1 & 2^1 & 3^1 \\ 1^2 & 2^2 & 3^2 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 3 \\ 1 & 4 & 5 \end{bmatrix}$$

- F D = C
  - where D is the vector of three data words
  - C is the vector of the three parity words
- Store D and C on the disks

Complexity of Reed-Solomon

- Encoding
  - Time: O(k n) GF[2^w]-operations for k check words and n disks
- Modification
  - like Encoding
- Decoding
  - Time: O(n^3) for matrix inversion
- Ease of implementation
  - check disk overhead is minimal
  - complicated decoding

DAAD Summerschool Curitiba 2011

Aspects of Large Scale High Speed Computing Building Blocks of a Cloud

Storage Networks

2: Virtualization of Storage: RAID, SAN and Virtualization

Christian Schindelhauer
Technical Faculty
Computer-Networks and Telematics
University of Freiburg