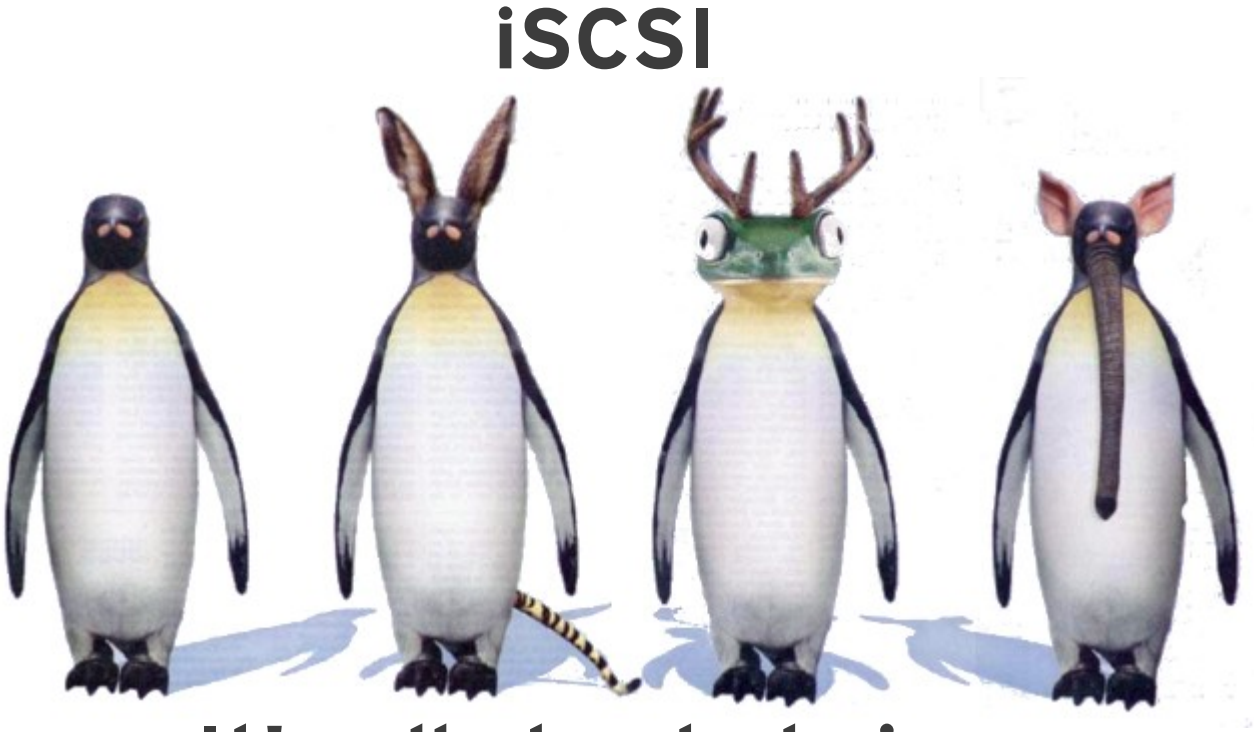
iSCSI - „Best Practice“ und Optimierung



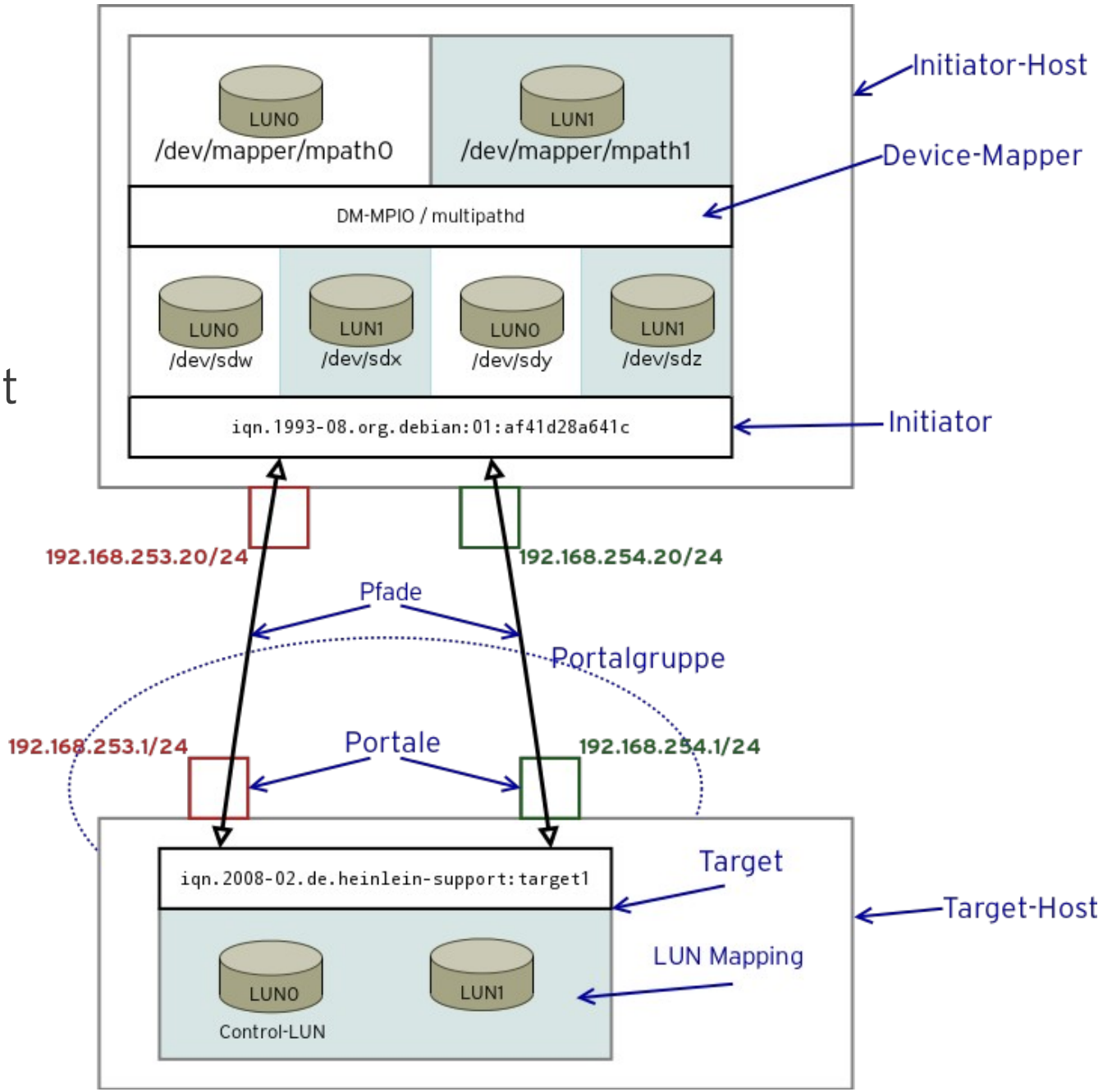
Wer sind wir?

- wir bieten seit 20 Jahren Wissen und Erfahrung rund um Linux-Server und E-Mails
- IT-Consulting und 24/7 Linux-Support mit 21 Mitarbeitern
- Eigener Betrieb eines ISPs seit 1992
- Täglich tiefe Einblicke in die Herzen der IT aller Unternehmensgrößen

heinlein

iSCSI

- Begriffsübersicht
- schematischer Aufbau



Stephan Seitz <s.seitz@heinlein-support.de>

Teil 1:

Der Protokoll-Stack

Stephan Seitz <s.seitz@heinlein-support.de>

SCSI over TCP/IP over ... ?

- iSCSI stellt hohe Ansprüche
  - Geringe Latenz
  - Hoher Durchsatz
  - Reagiert sehr sensibel auf
    - Packet Re-Ordering
    - Fragmentierung
    - Packet Loss bzw. Re-Send
- over anything ?
  - Nicht sehr empfehlenswert
  - WAN ? Möglich, aber Routing (Koaleszenz, evtl. asymetrisch), nicht ideale Path-MTU
  - Ethernet (oder besser, z.B. Infiniband)

Stephan Seitz <s.seitz@heinlein-support.de>

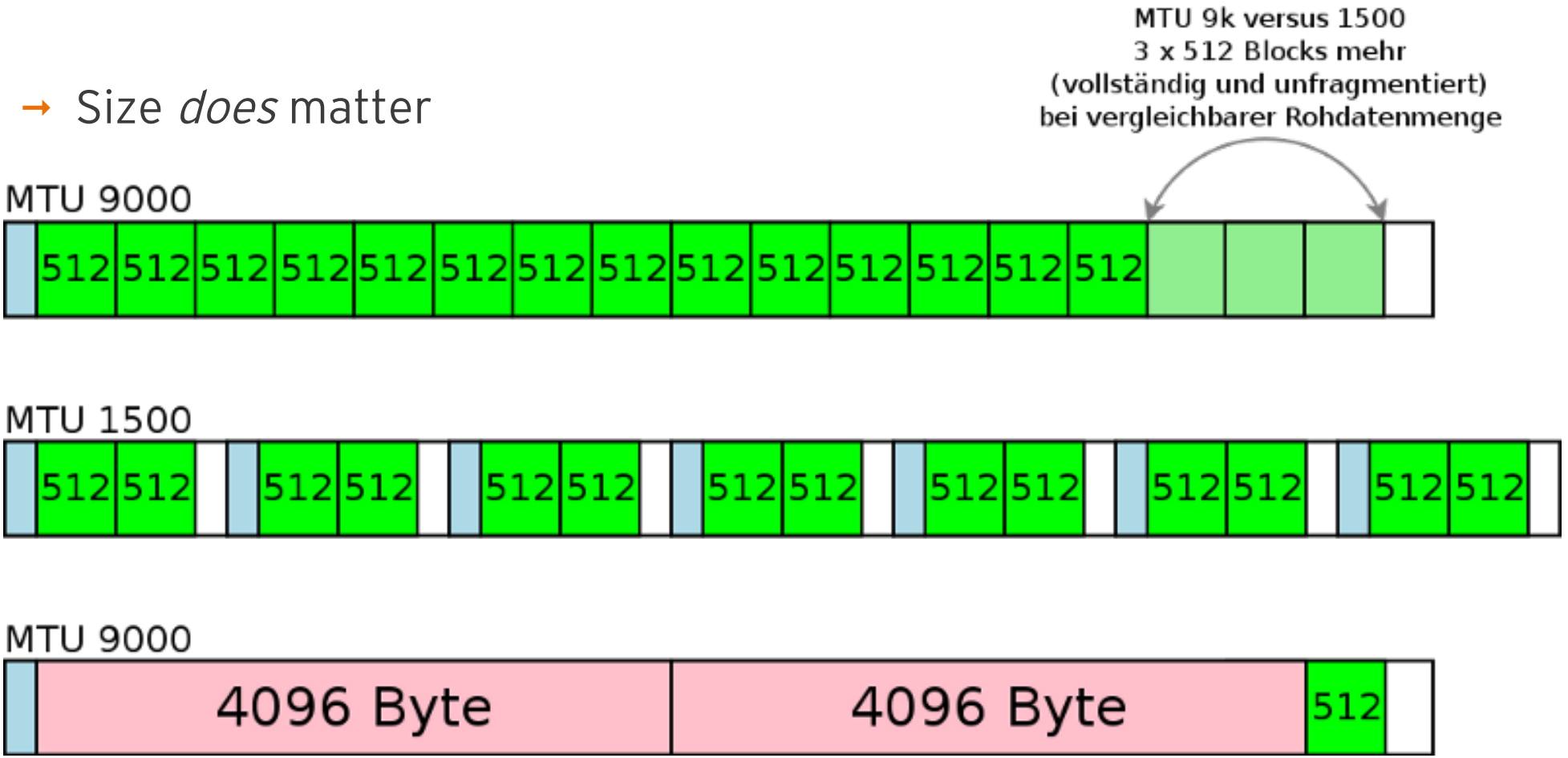
SCSI over TCP/IP over ... ?

- iSCSI stellt hohe Ansprüche
  - Geringe Latenz
  - Hoher Durchsatz
  - Reagiert sehr sensibel auf
    - Packet Re-Ordering
    - Fragmentierung
    - Packet Loss bzw. Re-Send
- over anything ?
  - Nicht sehr empfehlenswert
  - WAN ? Möglich, aber Routing (Koaleszenz, evtl. asymetrisch), nicht ideale Path-MTU
  - Ethernet (oder besser... , z.B. Infiniband)

Stephan Seitz <s.seitz@heinlein-support.de>

SCSI over TCP/IP over Ethernet

- Size *does* matter



SCSI over TCP/IP over Ethernet

- Jumboframes aktivieren

```
# ifconfig eth0 mtu 9000
```

- Komplettes Netzsegment beachten
  - MTU =*Maximum* Transmission Unit ... d.h. Der kleinste im Wert im Segment wird verwendet.

Stephan Seitz <s.seitz@heinlein-support.de>

Teil 2:

Hardware und Treiberunterstützung

Ethernet - Ein Blick auf die Hardware

- Hostadapter
  - iSCSI-Offload-Engine „iSCSI-HBA“?
    - Vorsicht! Häufig nicht mit 9k Jumboframes einzusetzen Datenblatt beachten
  - TCP/IP-Offload-Engine
    - Spart Interrupts, DMA-Operationen und CPU-Zyklen
  - I/OAT (Nur Intel)
    - Günstigeres Design als „klassisches“ TCP/IP-Offloading
  - 10GBase-T ?
    - Definitiv bei existierender 10GE-Infrastruktur
    - Requirements im Auge behalten
    - Im Vergleich zu 8GBit/s FC möglicherweise teurer
    - Stromverbrauch der Komponenten einkalkulieren

Ethernet - Ein Blick auf die Hardware

- Hostadapter
  - ethtool: *Soweit möglich, SG, TSO, GSO, GRO, LRO aktivieren.*

```
# ethtool -k eth0
Offload parameters for eth0:
scatter-gather: off ← Vectored I/O aktivieren
tcp-segmentation-offload: off ← Offload aktivieren
udp-fragmentation-offload: off ← nicht notwendigerweise
generic-segmentation-offload: off ← Offload aktivieren
generic-receive-offload: on ← Offload ist aktiv
large-receive-offload: off ← Offload aktivieren
```

  - Beispiel Intel igb Driver: *ioatdma muss vor igb geladen werden.*

```
# cat /etc/modprobe.d/igb.conf
softdep igb pre: ioatdma
options ioatdma ioat_dca_enabled=1
```

iSCSI Infrastruktur - open-iscsi Initiator

- open-iscsi Optimierungsmöglichkeiten
  - Unterstützt der verwendete NIC (hier besser: HBA) iSCSI Offloading ?
    - Deviceunterstützung testen (9k Frames). Benchmark-Vergleich!
    - Aktuell implementiert sind:
      Chelsio T3 10GBit Adapter (kmod: **cxgb3**, iscsiadm: **cxgb3i**)
      Broadcom NetXtremeII 1GBit Adapter (kmod: **bnx2**, iscsiadm: **bnx2i**)
      Broadcom NetXtremeII 10GBit Adapter (kmod: **bnx2x**, iscsiadm: **bnx2i**)
  - Via iser auch Unterstützung für iSCSI over Infiniband.

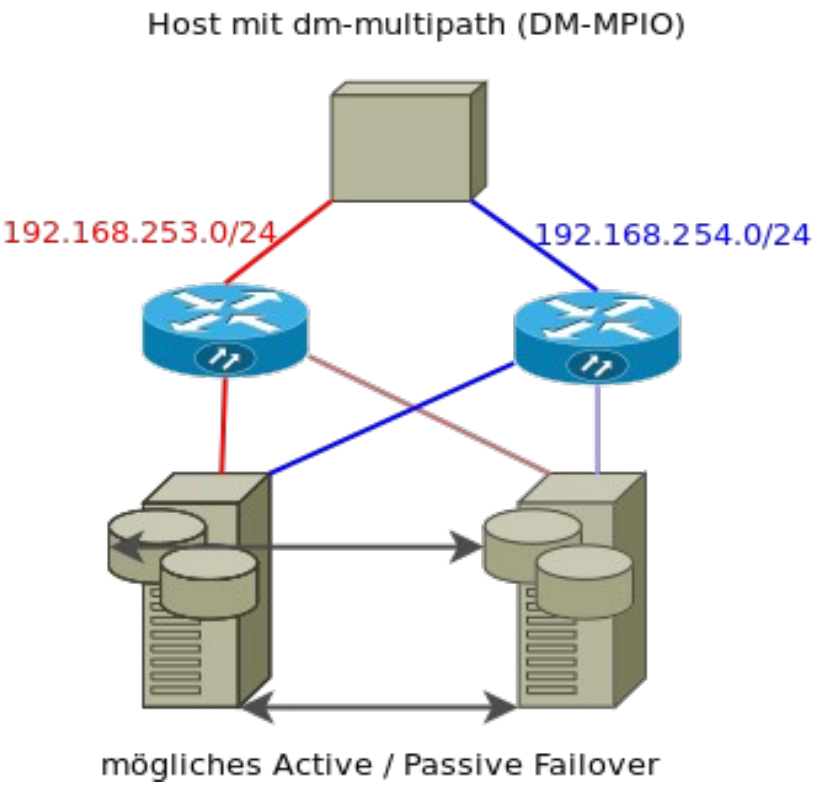
Stephan Seitz <s.seitz@heinlein-support.de>

Teil 3:

Multipathing

iSCSI Infrastruktur - MPIO

- Vorteile von Multipathing (DM-MPIO)
  - Path-Failover
  - Je nach Storage auch Controller-Failover
  - **`queue_if_no_path`** möglich
    g*efährlich, wenn Pfade dauerhaft offline sind*
  - Path-Selector für diverse Storages
    - SCSI-3 ALUA
    - EMC (Proprietär)
    - NetApp (Proprietär)
    - LSI RDAC
  - Konsequenzen
    - Höhere Ausfallsicherheit
    - Bessere Lastverteilung



ISCSI Infrastruktur - MPIO

- Multipath Caveats und Optimierungsmöglichkeiten
  - **rr_min_io** = IOPS per Path (default **1000**, güenstigeres Round-Robin-Verhalten bei deutlich kleineren Chunks, je nach Infrastruktur **20-30**)
  - **queue_without_daemon** (default **yes**, verhindert bei z.B. Shutdown oder Reboot sauberes Deaktivieren des /dev/mpath Devices wenn iSCSI bereits down ist. → **no**)
  - **path_checker** (Generisch **directio** oder **readsector0**. Verursacht zwar kleine, aber unnötige Leseoperation. Je nach Storageunterstützung **tur**, **rdac**, **hp_sw** oder **emc_clariion** verwenden)
  - **prio_callout** (Generisch **none**, wenn durch das Storage unterstützt **mpath_prio_emc**, **mpath_prio_alua**, etc. bevorzugen)
  - **path_grouping_policy** (Default **multibus,** bei entsprechender **prio_callout** Unterstützung ist **group_by_prio** bevorzugen)
  - Achtung bei **queue_if_no_path**. Sicherstellen, dass beim Wegfall aller Pfade eine schnelle Reaktion stattfindet. Ansonsten besser **no_path_retry** verwenden.

Stephan Seitz <s.seitz@heinlein-support.de>

Teil 4:

Storage bzw. iSCSI-Targets

ISCSI-Targets - Kaufen oder selbst aufsetzen?

- Wahl des / der Targets
  - Häufig bereits vorhanden, möglicherweise zusätzliche iSCSI-Portale an bestehenden „grossen“ FC-SAN
  - Definierte Ziele helfen bei der Auswahl
- Appliance
  - Support und SLA, Ausrichtung allerdings häufig auf Windows oder VMWare ESX
  - Evtl. Features die u.U. nicht trivial selbst umzusetzen wären
    - Thin Provisioning
    - Multi Tier Cache Strategien
    - Online Deduplizierung
  - u.U. „Spezialitäten“ berücksichtigen. Bspw. Nur eine virtuelle Portal-IP bei Equalogic

Stephan Seitz <s.seitz@heinlein-support.de>

iSCSI Targets unter Linux - Die Qual der Wahl.

- STGT http://stgt.sourceforge.net/
  - Ehem. Kernel In-Tree <= 2.6.20, wurde 2010 zugunsten LIO aufgegeben
- LIO http://www.linux-iscsi.org/
  - aktuell Kernel In-Tree >=2.6.38, Linuxphilosophie (ConfigFS)
  - ALUA Support
  - Target-Portal-Group LUN-Mapping
- SCST http://scst.sourceforge.net/
  - Out-Of-Tree aber sehr gute Patch-Chain
  - Neben iSCSI gute FC Unterstützung, IBM vSCSI, SRP Support
  - Häufig in Linux Appliances verwendet, performanteste Implementierung (12/2012)
- IET http://iscsitarget.sourceforge.net/
  - Funktional, aber aktuelle Features fehlen, nicht sehr auf Performance ausgerichtet

iSCSI Targets - generelle Optimierungen

- RAID bzw. viel hilft viel
  - Generell: Ja! - Intelligent Kombiniert: Noch mehr...

Level	Write	Read	Chunksize	Günstige Anzahl	Zweck
1 bzw. 10	n/2	n	grösser	mehr :)	Random Read/Write
5 bzw. 50	1 bis n-1	n-1 bis n	n=3 kleiner n>=8 grösser	>=5	Grosse Daten, Read
6 bzw. 60	1 bis n-2	n-2 bis n-1	Dito. mehr Spindeln grössere Chunks	>=8	Wie 5 bzw. 50

  - Sehr guter Artikel auf Wikipedia: http://de.wikipedia.org/wiki/RAID
  - Finger weg von Host-RAID (Fake-RAID), hier besser gleich Software-RAID
  - Bei Hardware-RAID: Kosten für Reserve Controller einplanen. Consumer-Hardware ist hier ungeeignet.

iSCSI Targets – generelle Optimierungen

- Den Blick aufs Ganze richten
  - Unter Idealbedingungen: Chunk- und Stripe-Size an den zu erwartenden Workload bzw. die Blockgrösse des iSCSI-Targets und die inode-Grösse des Dateisystems anpassen
  - Unter Realbedingungen: Extreme vermeiden.
- Alignment beachten
  - Beginnt je nach Aufbau bereits im Storage
  - Chunk-Size → HW-Blocksize → Partition(?) → PV /VG → LV → iSCSI-blockio
    bzw.
    Chunk-Size → HW-Blocksize → Partition(?) → FS → ImageFile → iSCSI-fileio
  - Ungünstiges Block-Alignment kann schon im Storage die effektiven IOPS auf die Hälfte reduzieren!
  - In aktuellen Distributionen beherrschen fdisk, dm-raid und lvm auto-align bzw. „auto-offset“ . Die vorgegebenen Werte sind gut und lassen sich händisch nur verschlechtern.

iSCSI Targets – generelle Optimierungen

- Differenzierung von File-I/O und Block-I/O
  - File-I/O
    - Operationen (open(), read(), write(), append(), ...) werden über die File API des Betriebssystems abgebildet.
    - Verwendet Virtual Memory Pages
    - Interaktion mit dem Betriebssystem bzw. Dateisystem
  - Block-I/O
    - Operationen werden direkt zur Low-Level-API durchgereicht.
    - Interaktion mit dem jeweiligen Device-Scheduler
  - Nur für das Management zwischen Device und Target. Logisches Target ist immer ein Block-Device.

iSCSI Targets - generelle Optimierungen

- File-I/O versus Block-I/O

	File-I/O RAM als Data-Cache	Block-I/O Max. Command-Cache
Read-Performance	o	o
Write-Performance	+	o
Re-Read-Performance	+ +	o
Failsave	- -	+

  - Je nach Implementierung: Verschiedene „Flavours“ möglich (DIRECT_IO)

iSCSI Targets unter Linux - Optimierungen

- Device-I/O Scheduler
  - Beispiel für **cfq** („completely fair queue“ - häufiger Standard)

```
# HWBS = $(cat /sys/block/sdzz/queue/max_hw_sectors_kb)
# echo $HWBS > /sys/block/sdzz/queue/max_sectors_kb
# /sbin/blockdev -setra $HWBS /dev/sdzz

# echo "cfq" > /sys/block/sdzz/queue/scheduler
# echo 256 > /sys/block/sdzz/device/queue_depth
# echo 512 > /sys/block/sdzz/queue/nr_requests
# echo 0 > /sys/block/sdzz/queue/iosched/slice_idle
```

  - Beispiel für **deadline** (Requests werden nach Ablauf einer Zeit verworfen)

```
# echo "deadline" > /sys/block/sdzz/queue/scheduler
# echo 4 > /sys/block/sdzz/queue/iosched/fifo_batch
# echo 150 > /sys/block/sdzz/queue/iosched/read_expire
# echo 1500 > /sys/block/sdzz/queue/iosched/write_expire
# echo 128 > /sys/block/sdzz/queue/read_ahead_kb
```

iSCSI Targets unter Linux - Optimierungen

- Device-I/O Scheduler
  - Je nach eingesetzter Hardware

```
1 = cpu group affinity, 2 = cpu core force affinity
# echo 1 > /sys/block/sdzz/queue/rq_affinity
```

  - http://doc.opensuse.org/products/draft/SLES/SLES-tuning_sd_draft/cha.tuning.io.html
- Memory, GC-Verhalten, CPU-Affinität - evtl. nur Homöopathie

```
# sysctl vm.dirty_ratio=40 # Default 10 (% RAM) (+)
# sysctl vm.dirty_background_ratio=15 # Default 5 (+)
# sysctl vm.vfs_cache_pressure=60 # Default 100 (-)

# sysctl kernel.sched_nr_migrate=64 # Default 32 (+)
# sysctl kernel.sched_migration_cost_ns=400000 # (-)
```

  - http://doc.opensuse.org/products/draft/SLES/SLES-tuning_sd_draft/cha.tuning.taskscheduler.html

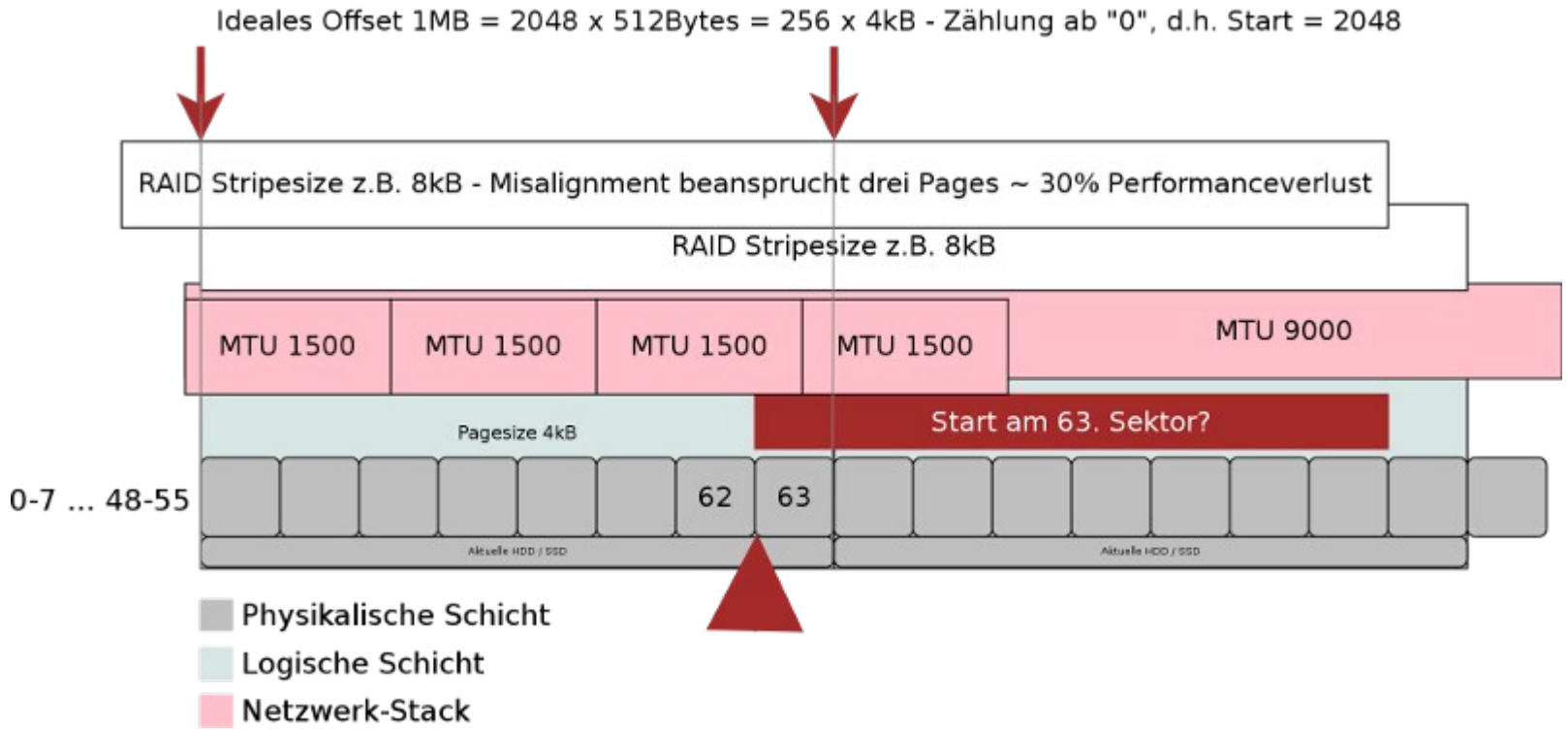
Stephan Seitz <s.seitz@heinlein-support.de>

Teil 5:

Generelle Betrachtung

iSCSI - generelle Optimierungen

- Alignment beachten



Stephan Seitz <s.seitz@heinlein-support.de>

iSCSI - generelle Optimierungen

- Alignment beachten
  - Endet nicht am iSCSI-Target.
  - iSCSI-LUN → HW-Blocksize → Partition(?) → PV /VG → LV → ?
    bzw.
    iSCSI-LUN → HW-Blocksize → Partition(?) → FS → ?
  - Ungünstiges Block-Alignment im Initiator-Host reduziert nicht nur drastisch die möglichen IOPS sondern belastet auch den iSCSI-Pfad unnötig.
  - Déjà-vu ... In aktuellen Distributionen beherrschen fdisk, dm-raid und lvm auto-align bzw. „auto-offset“ . Die vorgegebenen Werte sind gut und lassen sich händisch nur verschlechtern. Bei echten Long-Term (z.B. RHEL 5): Faustregel 1MB Offset
- Vernünftiges Tiering
  - Wenige, für unterschiedliche Workloads ausgelegte LUN skalieren meist besser als massenhafte, undifferenziertere LUN.

Stephan Seitz <s.seitz@heinlein-support.de>

iSCSI - generelle Optimierungen

- ISCSI Paketgrössen
  - **MaxBurstLength** (max. 2^24-1, nur 2er Potenzen gültig)
  - **MaxRecvDataSegmentLength** (max. 2^24-1, nur 2er Potenzen gültig)
    - Target und Initiator betrachten. Ähnlich wie MTU, der kleinere Wert zählt.
- TCP/IP
  - ifconfig txqueuelen (default 1000)
    Meinungen gehen weit auseinander (Stichwort: Buffer Bloat)
  - In homogenen, autarken Netzen, neben Anpassungen an WMEM und RMEM
    - net.ipv4.tcp_fastopen = 1
    - net.ipv4.tcp_sack = 1
    - net.ipv4.tcp_low_latency ist irrelevant. Besser rmem und wmem ausrechnen!

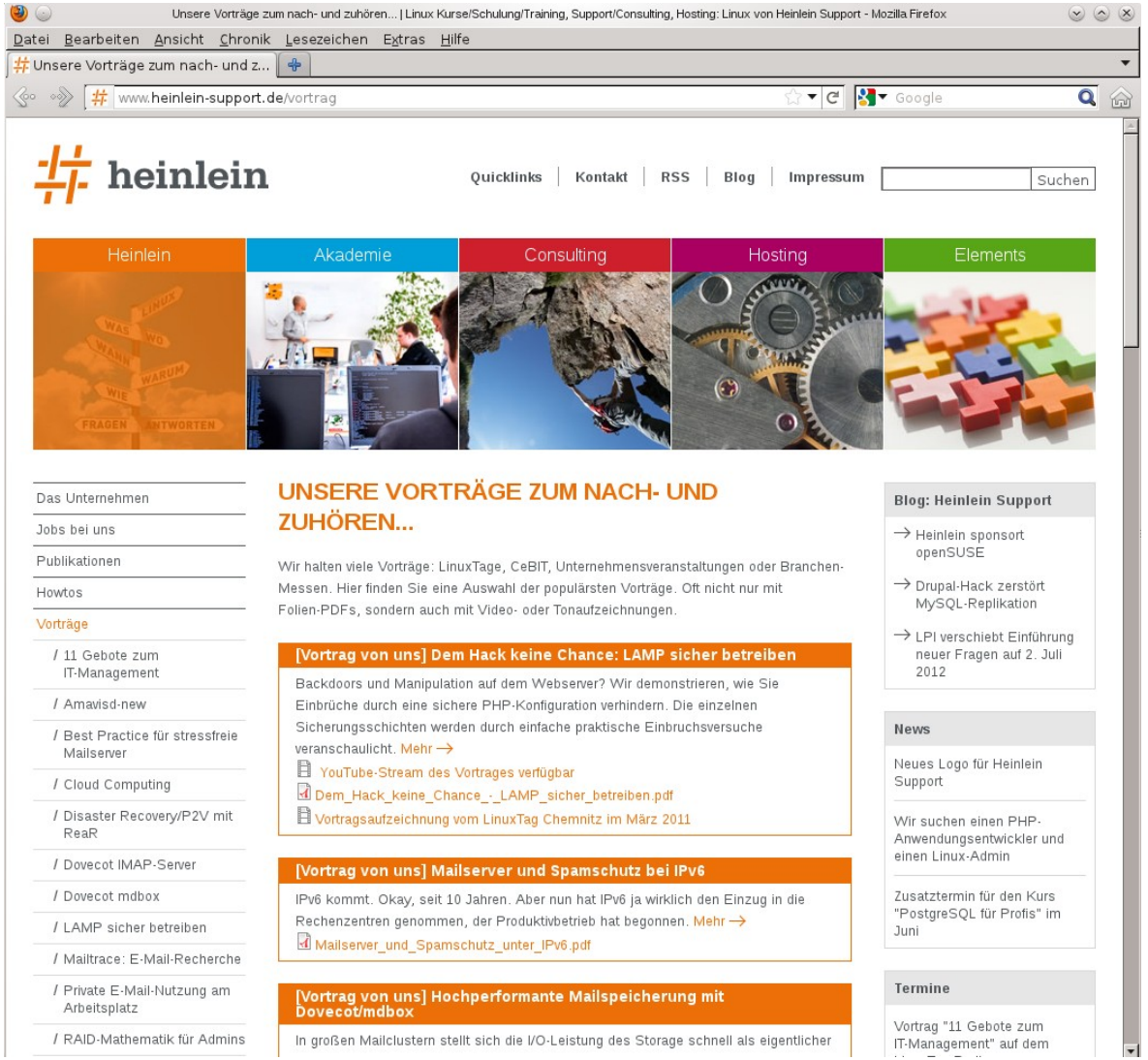
Stephan Seitz <s.seitz@heinlein-support.de>

Ein Blick über den Tellerrand: iSCSI ./. FCoE

- Anwendungsbereich wird sich vermutlich ausdifferenzieren
  - iSCSI setzt auf TCP/IP, FCoE auf FC-Frames mit eigenem Ethertype
  - Vorteil FCoE
    - FC-Frames bleiben erhalten, d.h. Direkte Kopplung mit FC-Netzen möglich
    - Geringerer Overhead im Vergleich zu iSCSI
  - Nachteile FcoE
    - Konvergente Netz-Infrastruktur notwendig, z.B. CNAs, Fabric-Extender
    - Hardwarekosten von FCoE vergleichbar zu FC

heinlein

Stephan Seitz <s.seitz@heinlein-support.de>



Ja, diese Folien stehen auch als PDF im Netz...
http://www.heinlein-support.de/vortrag

Stephan Seitz <s.seitz@heinlein-support.de>

Soweit, so gut.

Gleich sind Sie am Zug:
Fragen und Diskussionen!

Stephan Seitz <s.seitz@heinlein-support.de>

Wir suchen:
Admins, Consultants, Trainer!

Wir bieten:
Spannende Projekte, Kundenlob, eigenständige Arbeit, keine Überstunden, Teamarbeit

...und natürlich: Linux, Linux, Linux...

http://www.heinlein-support.de/jobs

Und nun...

- Vielen Dank für's Zuhören...
- Schönen Tag noch...
- Und viel Erfolg an der Tastatur...

Bis bald.

heinlein

Stephan Seitz <s.seitz@heinlein-support.de>

Heinlein Support hilft bei allen Fragen rund um Linux-Server

HEINLEIN AKADEMIE

Von Profis für Profis: Wir vermitteln die oberen 10% Wissen: geballtes Wissen und umfangreiche Praxiserfahrung.

HEINLEIN HOSTING

Individuelles Business-Hosting mit perfekter Maintenance durch unsere Profis. Sicherheit und Verfügbarkeit stehen an erster Stelle.

HEINLEIN CONSULTING

Das Backup für Ihre Linux-Administration: LPIC-2-Profis lösen im CompetenceCall Notfälle, auch in SLAs mit 24/7-Verfügbarkeit.

HEINLEIN ELEMENTS

Hard- und Software-Appliances und speziell für den Serverbetrieb konzipierte Software rund ums Thema eMail.