OSL RSIO

Remote Storage I/O

Storage-Networking der nächsten Generation

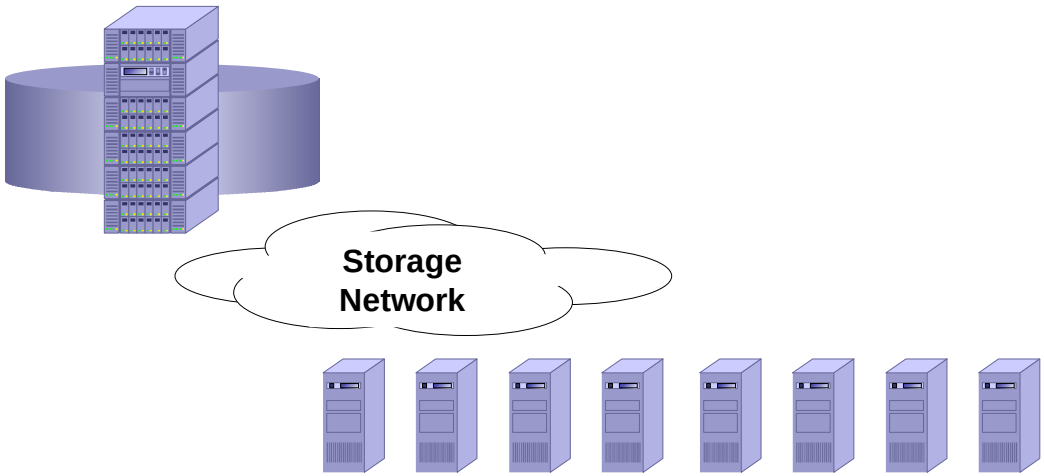
Bert Miemietz

OSL Gesellschaft für
offene Systemlösungen mbH

Warum RSIO ?

Speichernetzwerke

Heute Standard im Rechenzentrum - Warum?



- ***Spezialisierung / Funktionsteilung***
  - *Spezialsysteme für einfachere Handhabung, bessere Verfügbarkeit und Performance*
- ***Flexibilität***
  - *Trennung Compute Node – Storage erlaubt Anwendungsmobilität, HV, DR*
  - *Trennung ermöglicht diverse Virtualisierungsansätze*
- ***Heutige Massenspeichertechnologien sind langsam und fehleranfällig***
  - *Netzwerke versprechen Skalierbarkeit und bessere Verfügbarkeit*

Klassisch: Spezialisierte Netzwerke

Verschiedene Netzwerke für verschiedene Anwendungen

Aus Sicht des Anwenders sind sich beide Netzwerktechnologien heute im Verhalten oftmals ähnlicher als erwartet

Fibre Channel

- *Spezialprotokoll*
- *Block I/O*
- *Kanaleigenschaften*
- *Niedrige Latenz*
- *Hoher Durchsatz*
- *Niedrige CPU-Belastung*

- *NFS, SMB ...*
- *Backup*
- *IP over FC*
- *FC over IP*

Ethernet / IP

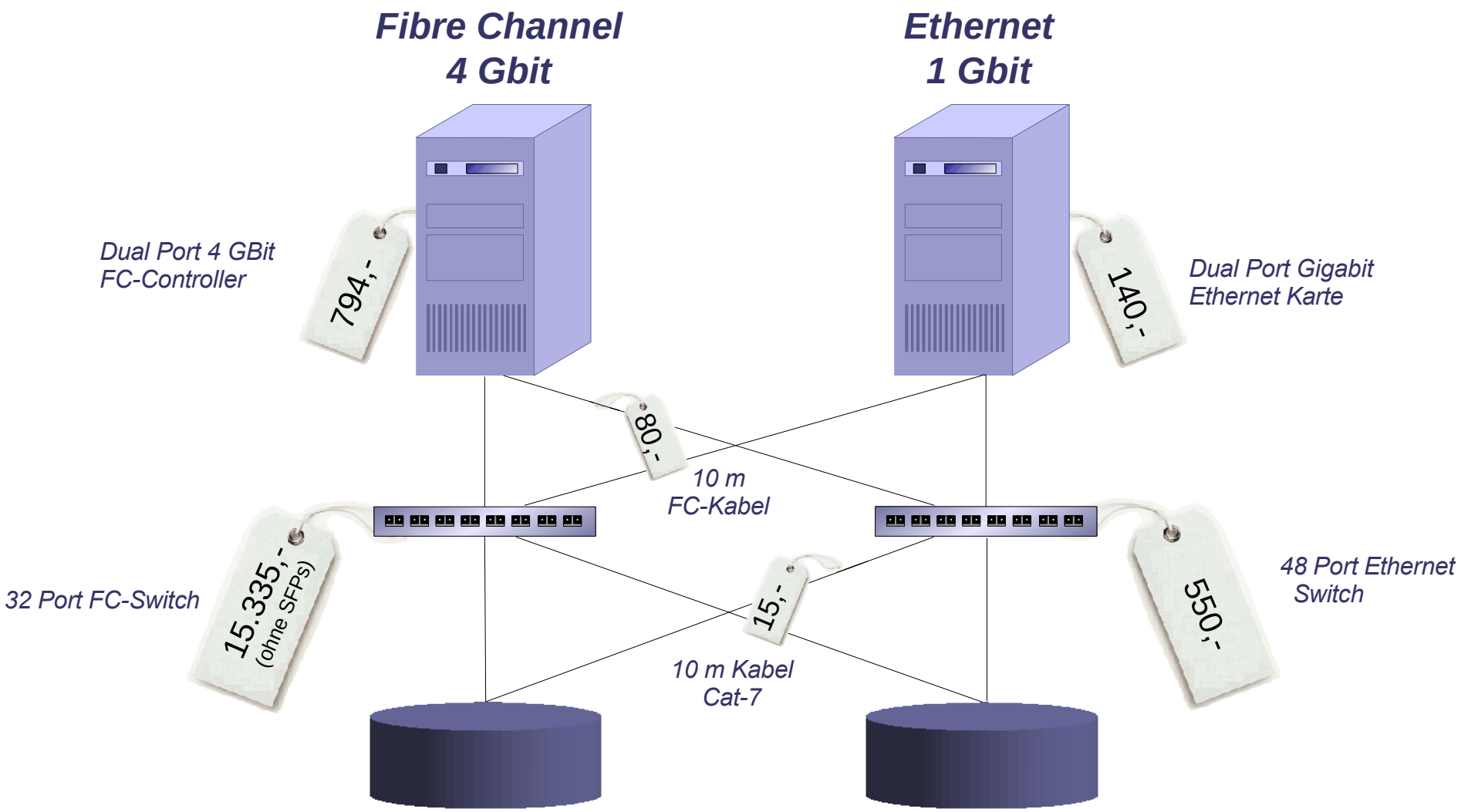
- *Universalprotokoll*
- *Primär von Applikationen getrieben*
- *Zweck: Kommunikation*
- *Client/Server-Applikationen*
- *Implementierung wesentlich im OS -> höhere CPU-Belastung*
- *Seit langem auch für Storage genutzt (NFS, Backup ...)*

Warum Storage über Ethernet?
Anforderungen und Möglichkeiten

- **Anforderungen und Erwartungen**
  - *Erfordernisse der Anwendungen und Protokolle (Kommunikation, Filesharing etc.)*
  - *Preisliche Motivationen*
  - *Einheitliche Infrastruktur, weniger Ports ?*
  - *Einfachheit, Flexibilität ?*
  - *Virtualisierungstechnologien, Verfügbarkeit von Treibern*
  - *Zusatzfunktionen (Konvertierungen, Filesystemsnapshots ...)*
- **Möglichkeiten**
  - *Gigabit-LAN heute vergleichsweise preiswert*
  - *Gigabit-LAN heute in vernünftiger Relation zur Geschwindigkeit einer Festplatte bzw. eines RAID-Systems*
  - *Gigabit-LAN heute in günstigerRelation zu Durchsatz-Anforderungen der Applikationen*
  - *Mehrere Gigabit-Ports je Server*
  - *Ethernet ist eigentlich (fast) kein Ethernet mehr -> Switching-Technologie*
  - *RAID-Systeme / Filer sprechen direkt die erforderlichen Protokolle*
  - *Neue Performance-Erwartungen an 10GBit-Ethernet*

Was ist mit den Kostenvorteilen?

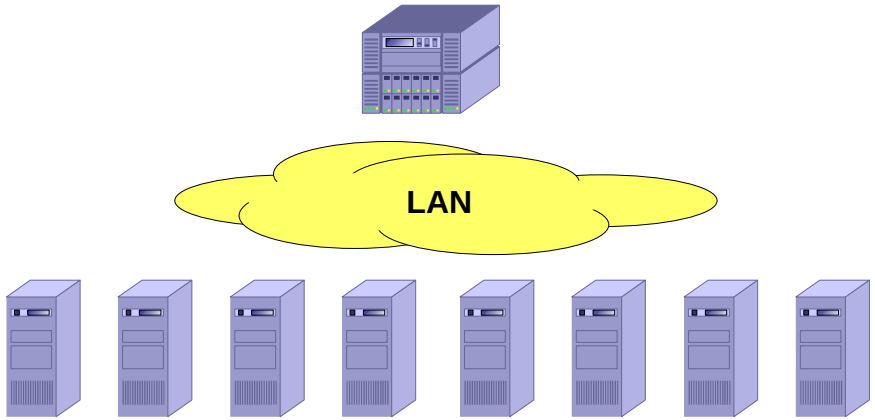
Ein Vergleich mit Fibre Channel (Stand 10/2009)



Storage über Ethernet heute: NFS, SMB, CIFS

NA(F)S präsentiert sich mit handfesten Vorteilen

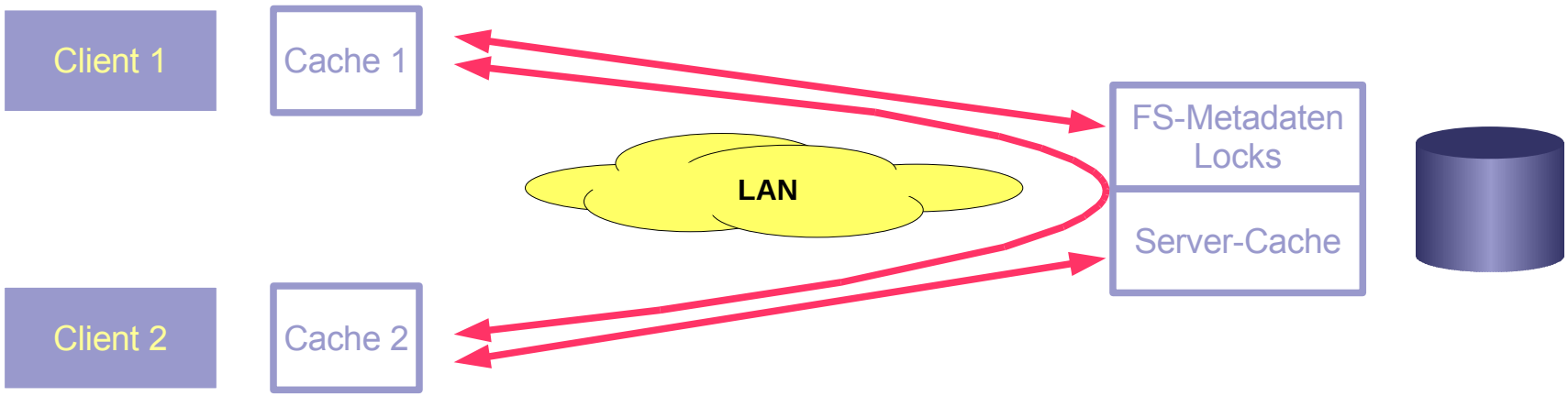
- *Spezialisierung auf Fileservices, dafür relativ einfache Handhabung*
- *kann komplexe RAID-Funktionen verbergen*
- *dateisystemtypische Funktionalitäten wie Snapshots und weitere Sonderfunktionen*
- *ermöglichen Filesharing*
- *weite Verbreitung und Unterstützung der wichtigsten Protokolle*
- *im Rahmen des heute Vorstellbaren Möglichkeiten nahezu ausgereizt*



Die Kehrseite des NAFS-Ansatzes

Es gibt auch prinzipbedingte Nachteile

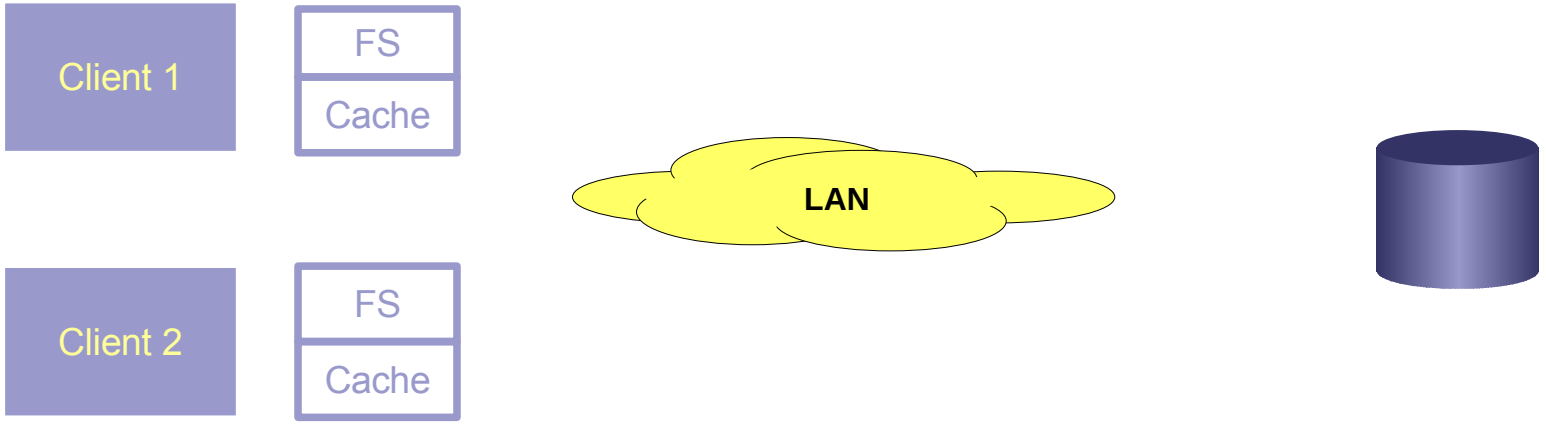
- *aufwendige Integration mit Server-OS (Zugriffskontrolle, User-Management)*
- *Cache- und Cohärenzproblematik, schwierige Nutzung der Client-Ressourcen*
- *nicht trivial: Skalierbarkeit, Hochverfügbarkeit, Multipathing*
- *feste Bindung an File-Access-Semantik*
- *mit zunehmender Funktionalität auch Zunahme von Komplexität und ggf. Inkompatibilitäten*



Starke Argumente für Block-I/O im RZ

Jenseits von Filesharing überwiegen die Vorteile

- *Volle Kontrolle des Client-OS über das Storage-Device*
- *Nutzbar für beliebige Filesysteme*
- *Keine Kopplung an Server-OS (Isolation, privates Identity Management)*
- *Nur Übertragung von I/O, nicht von Cache-Inhalten*
- *Cache liegt beim Client -> schnellster Zugriff, Client-Caches summieren sich auf*
- *Einfache Administration, schlankes Protokoll, hohe Geschwindigkeit*



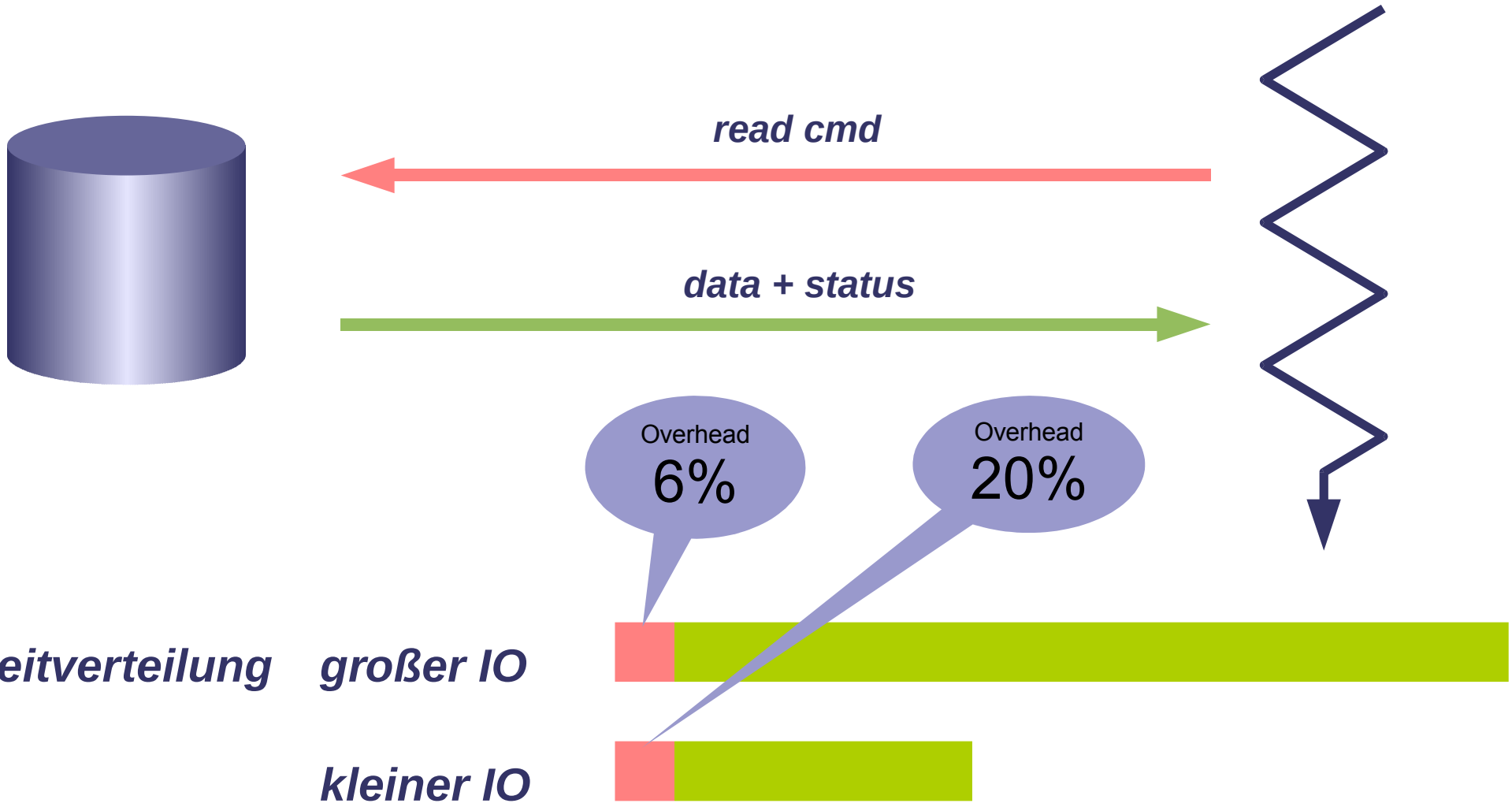
Block-I/O über Ethernet/IP - Performance

Performance hat viele Aspekte

- ***Latenz und Service Time***
  - *Relevant für einzelnen I/O und single threaded I/O*
- ***Maximaler Durchsatz***
  - *Relevant für multithreaded I/O*
- ***Skalierbarkeit / Parallelisierbarkeit***
  - *Thema für Multipathing*
- ***IO-Größe***
  - *Relevant für Nutzdatenanteil am Gesamtdurchsatz*

Bedeutung von Latenzen und IO-Größe

Heutige Paradigmen setzen Grenzen



Storage over Ethernet: Aktzeptable Latenzen?

Betrachtungen zur Performance

Zeit zur Übertragung von 32 Byte an 1 GBit-Lanboard

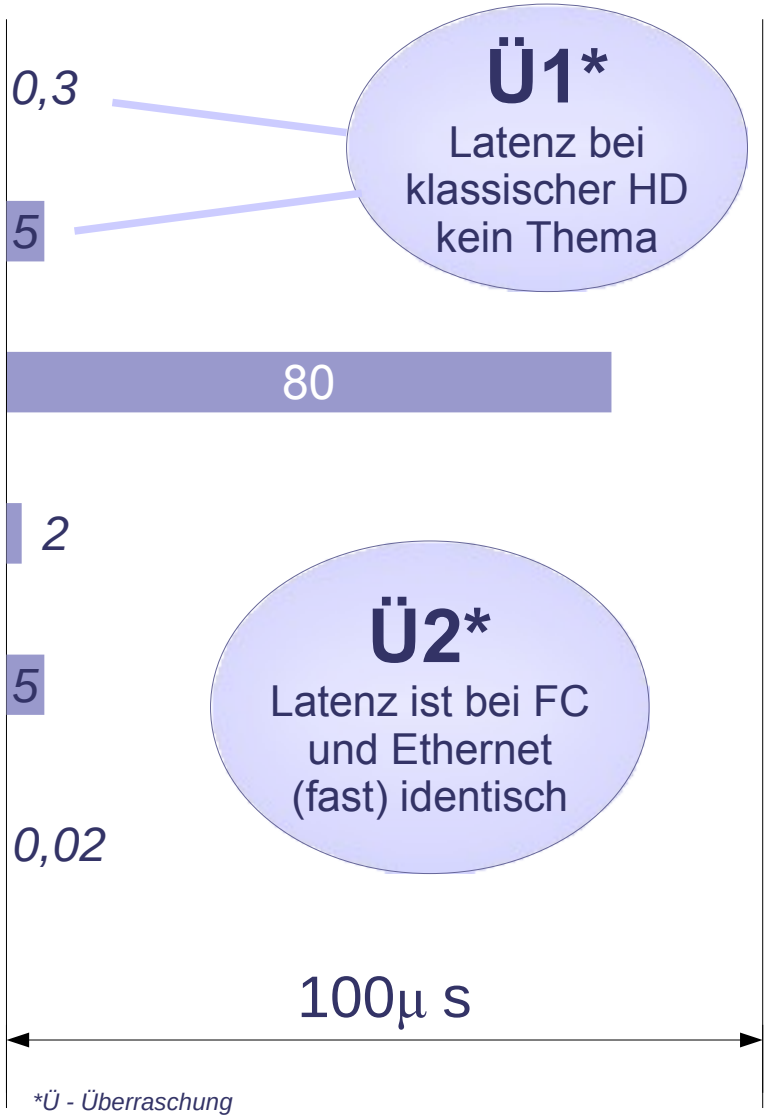
Zeit zur Übertragung von 512 Byte an 1GBit-Lanboard

Zeit zur Übertragung von 8kByte an 1GBit-Lanboard

memcpy 8k auf 600MHz-System

Threadwechsel per CV auf 600MHz-System

32Bit-Typkonvertierung auf 600MHz-System



*Ü - Überraschung

Storage over Ethernet: Aktzeptable Latenzen?

Betrachtungen zur Performance

Zeit zur Übertragung von 32 Byte an 1 GBit-Lanboard

Zeit zur Übertragung von 512 Byte an 1GBit-Lanboard

Es geht also:

Zeit zur Übertragung von 8kByte an 1GBit-Lanboard

1. Weniger um die Hardware.

2. Entscheidend um das Protokoll.

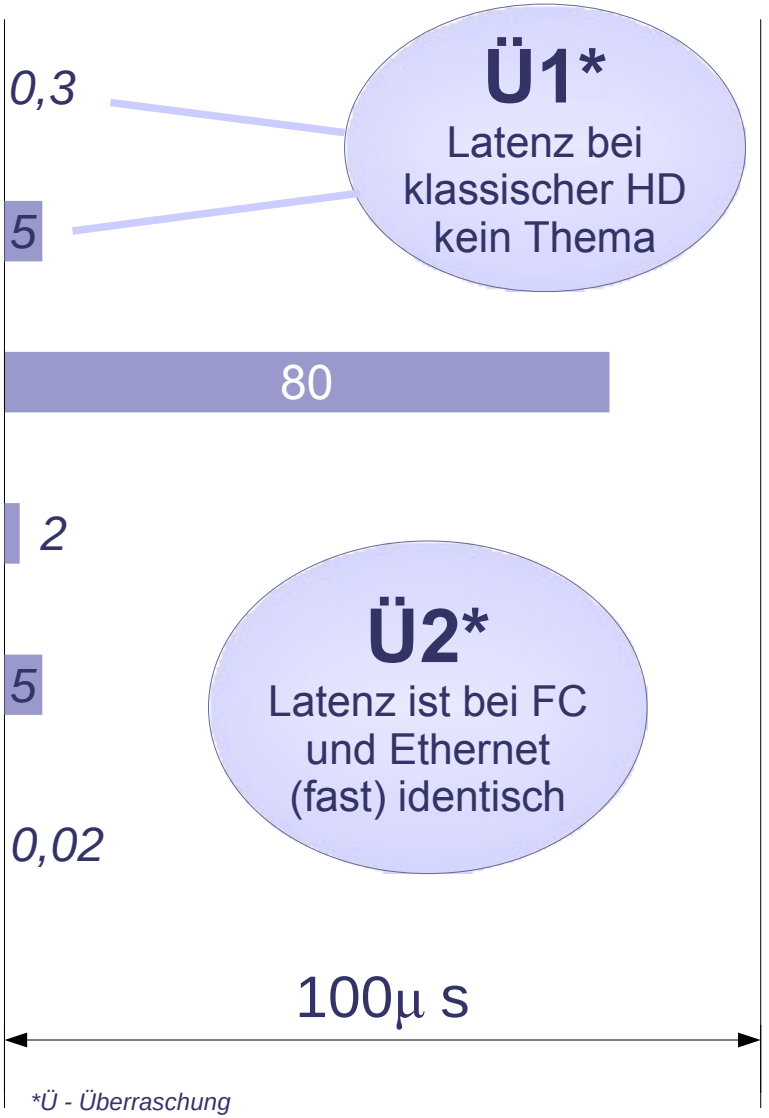
memcpy 8k auf 600MHz-System

3. Entscheidend um die Systemsoftware.

Threadwechsel per CV auf 600MHz-System

4. Nicht unerheblich um die Applikation.

32Bit-Typkonvertierung auf 600MHz-System



**Ü - Überraschung*

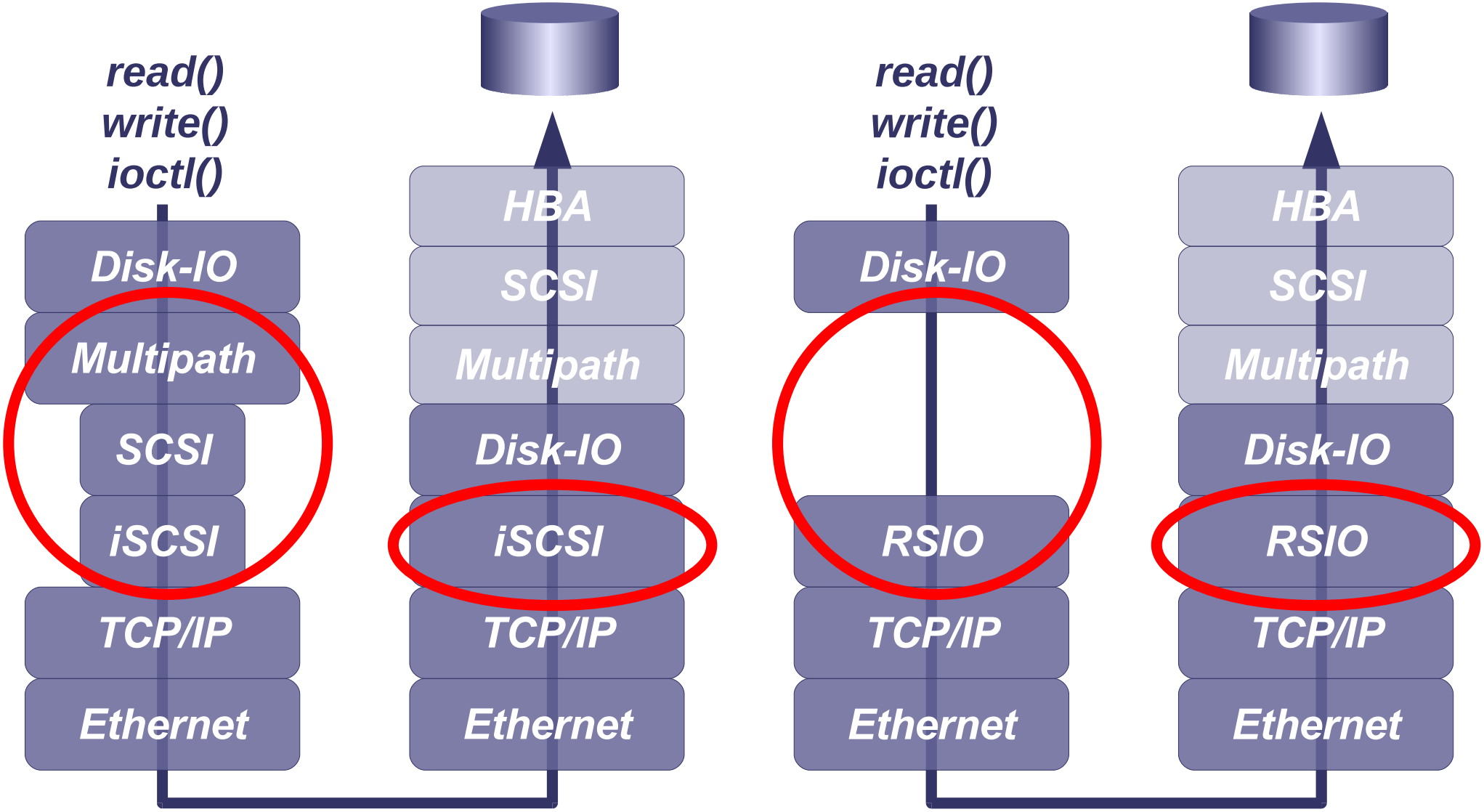
Block-I/O über Ethernet mit iSCSI

Bekanntes Protokoll über neues Medium

- *Low-Level-Protokoll auf IP umgesetzt*
- *Platfformunabhängig auf Server-Seite (Target)*
- *Starke Bindung an TCP, Offload-Engines auf Initiatorseite dennoch selten*
- *Tiefer Stack – nicht unerheblicher CPU-Bedarf*
- *Zahlreiche SCSI-Funktionen, aus Sicht der Anwendungen aber dennoch Verlust an Funktionalität*
  -> Storage-Management meist über andere Protokolle
- *Vernetzte, geclusterte Speichersysteme ohne speziellen Support*
- *Weitere Schwierigkeiten*
  - *Multipathing*
  - *Clustering / Parallelisierung*
  - *Nomenklatur*
  - *Target Portal Groups*

Block-I/O über Ethernet/IP – ohne SCSI

Ein anderer Ansatz mit OSL RSIO



RSIO - Remote Storage IO

Eckdaten der neuen Technologie für LAN-attached (shared) Block Devices

- *eigenes, von OSL entwickeltes Protokoll*
- *natürliche Erweiterung des Speichervirtualisierung auf LAN (Ethernet)*
- *voller Clustersupport möglich (Client und Server), vorrangig natürlich:*
- *Einbindung in OSL-Clustertechnologie*
- *alle relevanten IO-Aufrufe (read, write, ioctl ...) abbildbar*
- *hochportable Implementierung*
- *internes Layout berücksichtigt moderne CPU- und Serverkonzeptionen*
- *guter Durchsatz / gute Verfügbarkeit mit heutiger preiswerter Standardtechnik*
- *integrierte und bequeme Administration vom Host aus*
- *Erweiterbarkeit, Raum für intelligente IO-Lösungen*

Überblick zur Technologie

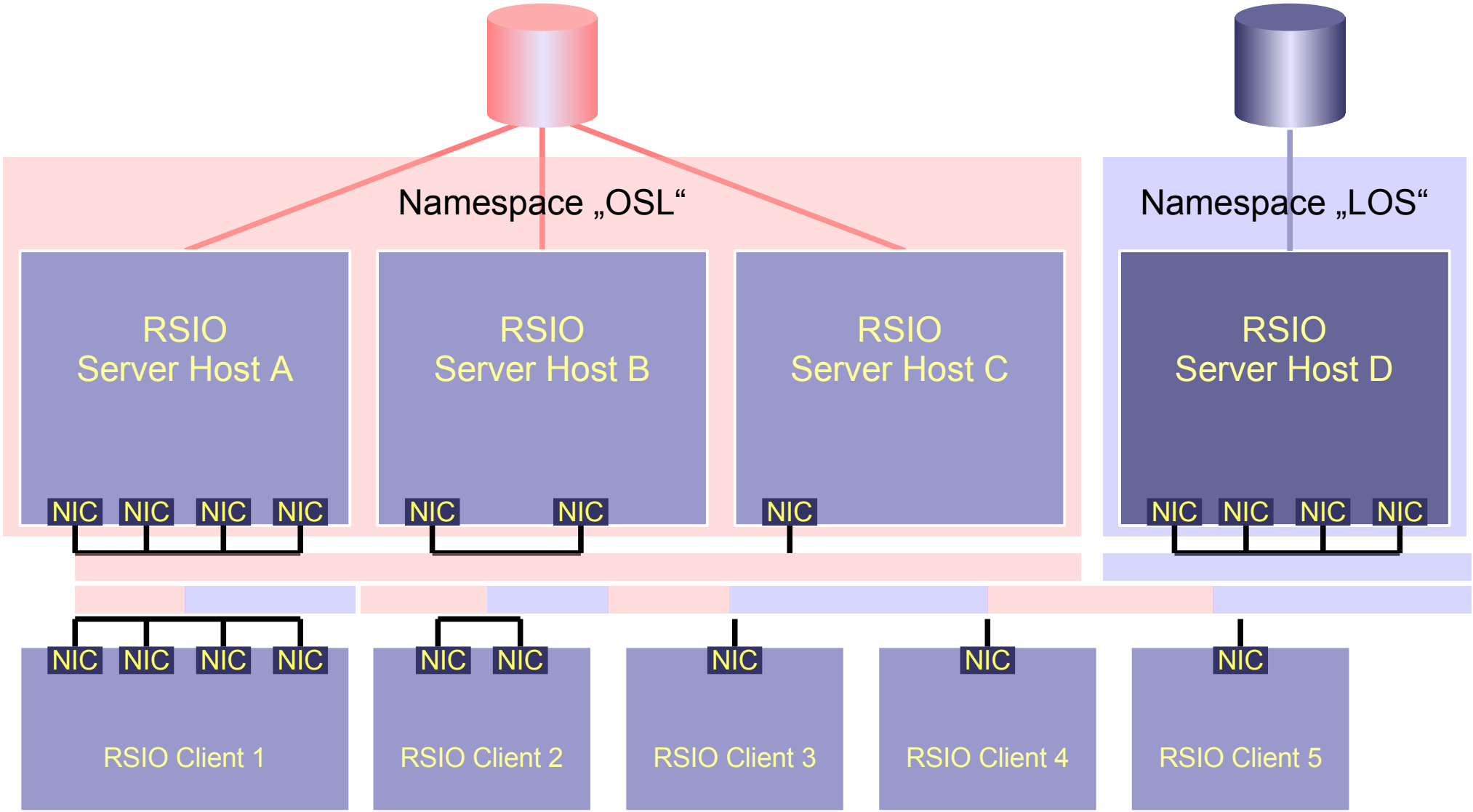
OSL Remote Storage IO (RSIO)

Etwas zu den Details der Implementierung

- *Definition eigener Frames*
  - *Unabhängigkeit von TCP*
  - *Durchsatz-Optimierungen*
  - *ermöglicht Zusatzfunktionen wie Checksum / Encryption*
  - *Frames mit variabler Größe*
  - *Overhead per Frame nur 16 Byte*
- *Trennung von Treiber und Transport*
  - *größerer Funktionsumfang bei hoher Portabilität*
  - *besseres Error-Handling*
  - *Performance offensichtlich kein Problem*
  - *hochflexibler Multithreading-Support*
  - *bessere Abschirmung des Kerns*
- *Integriertes autonomes Multipathing und Trunking*
- *Selbstkonfiguration und Error Recovery*
- *Unterstützung geclusterter Server*
- *vollständige Abbildung relevanter Schnittstellen möglich (z. B. dkio, mtio)*

RSIO - Architektur im RZ

Klar gegliedertes und flexibles administratives Konzept



Parameter der RSIO-Architektur

Flexible Client-Server-Implementierung

- *Ein Namespace definiert Server (und Clients) mit Zugriff auf dieselben Storage-Ressourcen*
- *Auf einem Serverhost können (nahezu) beliebig viele Server(prozesse) laufen*
- *Jeder Serverhost kann (nahezu) beliebig viele Clients bedienen*
- *jeder Client unterstützt den Zugriff auf bis zu 256 Server*
- *jede Maschine (Client und Server) unterstützt bis zu 8 Interfaces*
- *jeder Client hat simultan Zugriff auf verschiedene Namespaces*
- *Auto-Explorer*
  - *Ermitteln verfügbarer Verbindungen*
  - *Ermitteln der Schnittstelleneigenschaften*
  - *Test der Parameter auf der Übertragungsstrecke*

RSIO und Linux

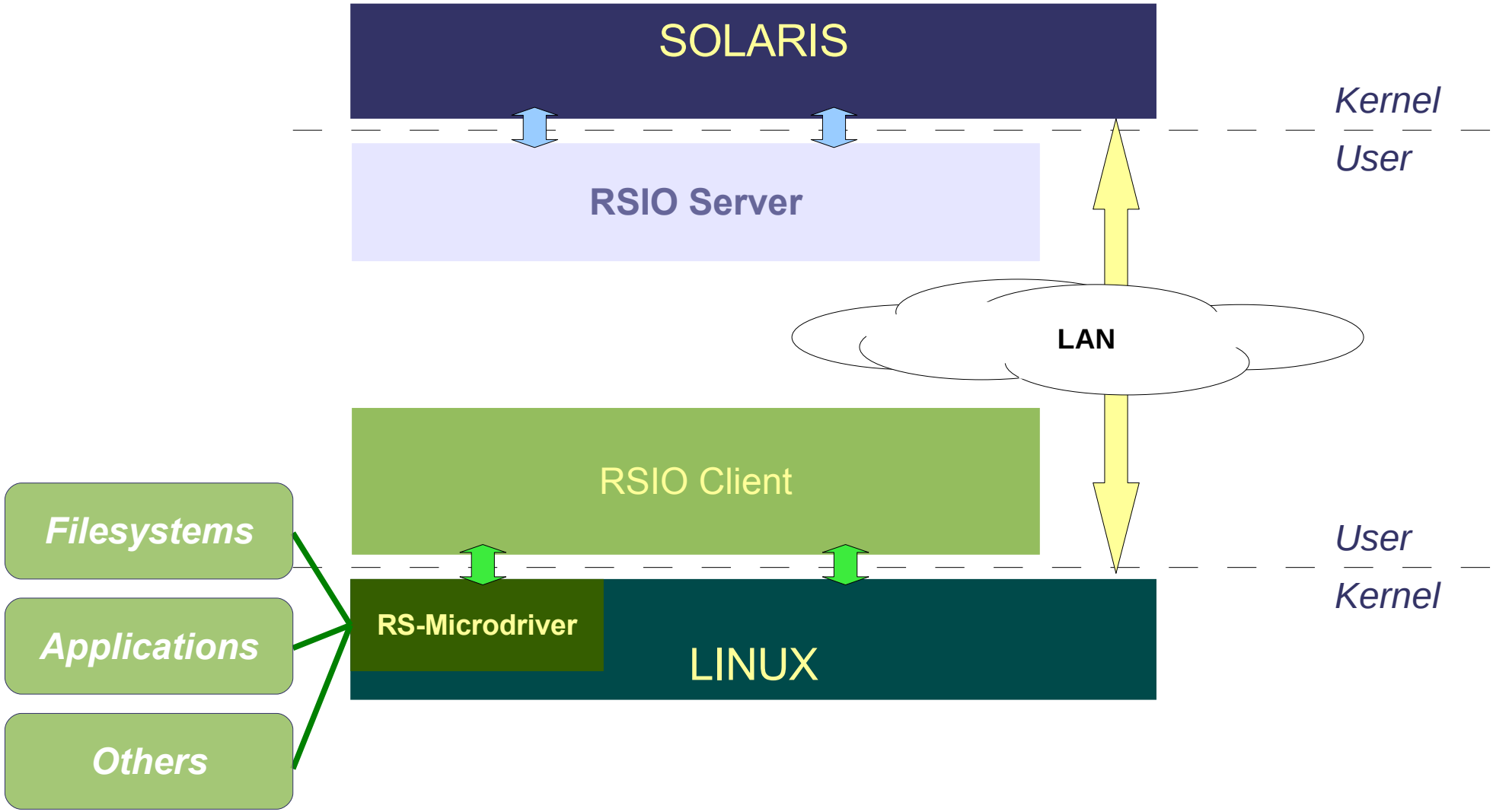
Und was ist mit Linux?

Was der technisch Interessierte zu diesem Thema wissen muß

- *GNU:*
  - ***gcc** sehr leistungsfähiges Werkzeug, auf vielen Plattformen verfügbar*
  - *Unterstützung aller für C relevanten Programmiertechniken und **Standards***
- *LINUX:*
  - *leistungsfähiges OS, verfügbar auf vielen Hardware-Plattformen*
  - *positiv formuliert: große Vielfalt (ubuntu, fedora, openSUSE, **debian**, Mandriva, LinuxMint, PCLinuxOS, **slackware**, gentoo linux, CentOS, **Red Hat**, SLES, SLED)*
  - *Gemeinsamkeit: Kernel + gcc/libc + System Calls, oft auch verfügbare Applikationen*
- *Kernel:*
  - *unbestritten leistungsfähig und hoch optimiert*
  - *keine stabilen internen Schnittstellen -> aktuelle Dokumentation?*
    -> Zukunftssicherheit für addon-Driver?
  - *Kernelprogrammierung sehr speziell bzw. proprietär (Kernel Build System, Macros etc.)*
  - *Konzepte unterscheiden sich z. T. deutlich von anderen Systemen (z. B. Multithreading)*
  - *Wartbarkeit des Kernels? (Strukturierung, Debugging ...)*
  - *Lizenzproblematik*

Wie umschiffen wir mögliche Probleme?

Die Entdeckung des Userspace...



Wie umschiffen wir mögliche Probleme?

Die Entdeckung des Userspace...

Vorteile:

1. Größter Teil läuft im Userspace
   - -> Portabilität
   - -> Systemstabilität
   - -> Fehlerbehandlung
   - -> Debugging
   - -> Handling (z. B. Clusterumgebungen)
   - -> Solaris kann vorrangige Entwicklungsplattform bleiben
2. Schlankes Kernelmodul und damit
3. Erleichterungen für die Portierbarkeit des Kernelmoduls

Und konkret ?
Die Umsetzung auf Linux

- **Entwicklungsplattform: SUSE Linux Enterprise Server 11 (x86_64)**
  - Kernel: 2.6.27.19-5-default

- **Ladbares Kernel-Modul (analog Solaris: “rs”)**
  - Modul heißt analog zu Solaris “rs”
  - Vorabinfo: `modinfo`
  - Laden: `insmod` (+ ggf. Optionen)
  - Anzeigen: `lsmod`
  - Entladen: `rmmod`

- **Besonderheiten**
  - Logging und Debug-System analog zu Solaris implementiert
  - sysfs-Integration, Management der Disk-Device-Nodes via `rsconfig`
  - Reservierung Major-Nummer für Block- und Char-Devices (Default: local 246)
  - Effektiv im Moment für Anwender nur Nutzung der Block-Devices (Disk)
  - modprobe-Konfiguration noch nicht enthalten

In's System geschaut

Die Sicht des Client-Administrators auf den Server

```
[root@big-6] rsconfig -q
000 osl
    clt: big-6
    srv: 000 big-5
         0  tvol1           disk         2097152 blocks of  512 bytes
         0  shadow          disk         2097152 blocks of  512 bytes
         0  z1_root         disk        10485760 blocks of  512 bytes
         0  sparse          disk        10485760 blocks of  512 bytes
         0  whole           disk        41943040 blocks of  512 bytes
         0  iscsit_cfg      disk           20480 blocks of  512 bytes
         0  target          disk         2097152 blocks of  512 bytes
         0  tconf           disk           20480 blocks of  512 bytes
         0  p07             disk       585920023 blocks of  512 bytes
         0  p08             disk       585920023 blocks of  512 bytes
         1  b07             disk       976545023 blocks of  512 bytes
         1  b08             disk       976545023 blocks of  512 bytes
         2  b09             disk       976545023 blocks of  512 bytes
         2  b10             disk       976545023 blocks of  512 bytes
         3  b11             disk       976545023 blocks of  512 bytes
         3  b12             disk       976545023 blocks of  512 bytes
[root@big-6] rsconfig -qv
000 osl (12345)
    clt: big-6 (0139dfX982)
    srv: 000 big-5 (id 1)
         0  tvol1           disk         2097152 blocks of  512 bytes
            c: /dev/av0/rtvol1
            b: /dev/av0/tvol1
```

In's System geschaut

Wie sich die Volumes auf dem Client darstellen

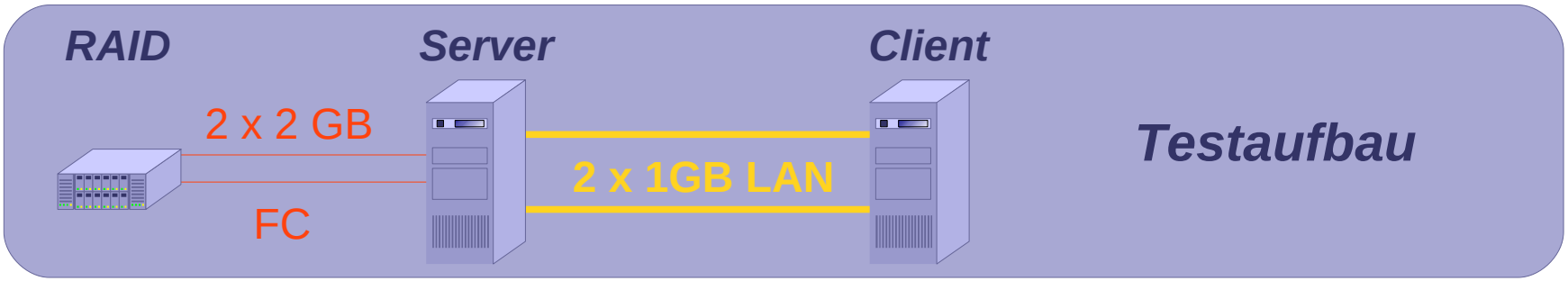
```
[root@big-6] rsconfig -lvv
osl:tvol1@0                                    2097152 blocks,   1 server(s)
    c: /dev/av0/rtvol1
    b: /dev/av0/tvol1

osl:shadow@0                                   2097152 blocks,   1 server(s)
    c: /dev/av0/rshadow
    b: /dev/av0/shadow

osl:z1_root@0                                 10485760 blocks,   1 server(s)
    c: /dev/av0/rz1_root
    b: /dev/av0/z1_root

osl:sparse@0                                  10485760 blocks,   1 server(s)
    c: /dev/av0/rsparse
    b: /dev/av0/sparse

osl:whole@0                                   41943040 blocks,   1 server(s)
    c: /dev/av0/rwhole
    b: /dev/av0/whole
```

Leistungsparameter

Und die Performance ?

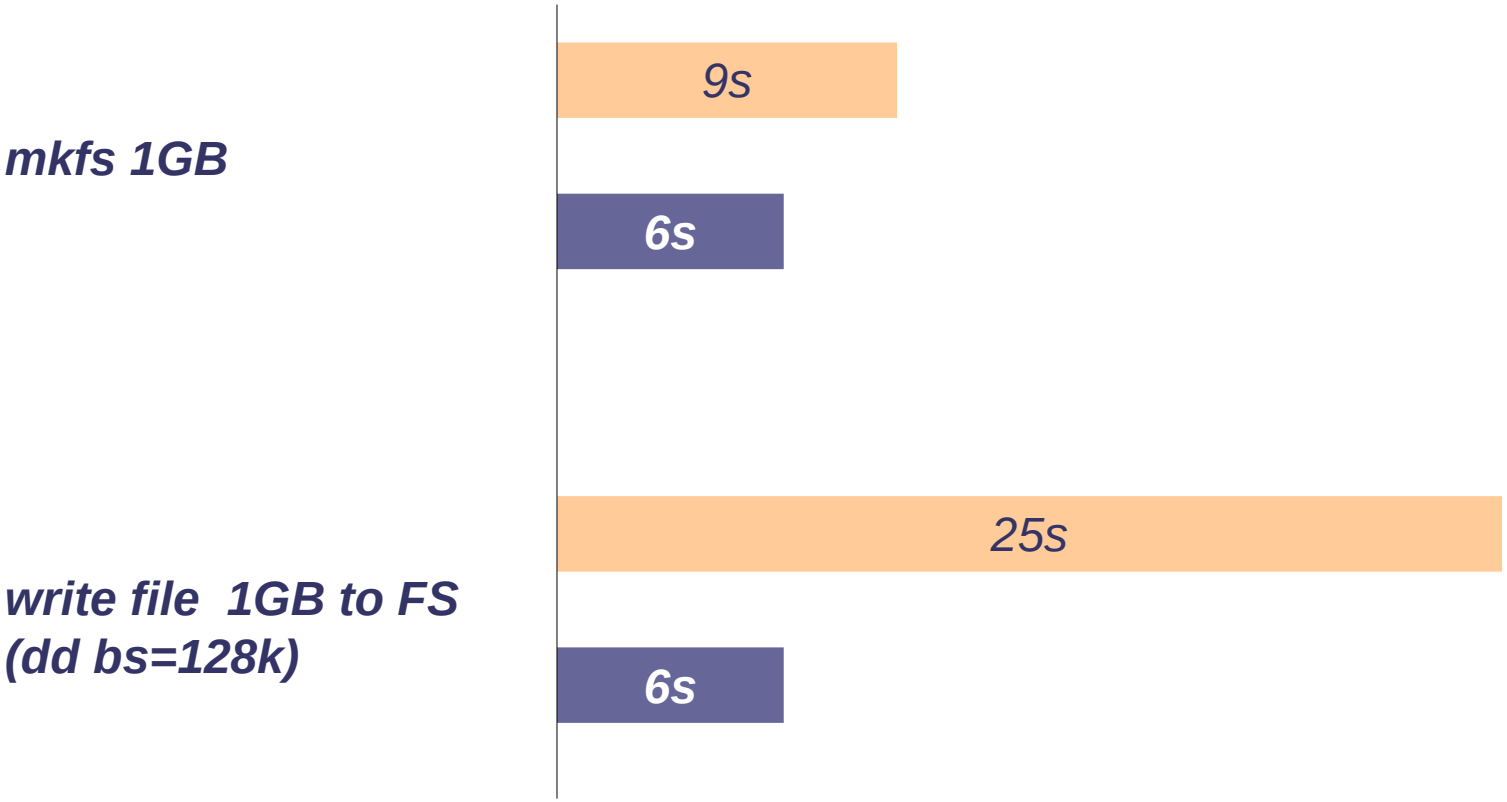
Nochmal kurz zur Theorie ...

- *wird zumeist schwächer sein als FC*
  - *schwächeres Universalprotokoll TCP/IP*
  - *bei IP werden erhebliche Kommunikationsteile im OS gerechnet*
  - *Ausnahmsweise stärker dort, wo der Server nicht über SAN/SCSI auf Storage zugreift*
- *Performance-Boost gegenüber NAFS durch IO-Vermeidung*
  - *virtueller IO-Cache bei exklusivem Zugriff*
  - *hier Vorteil gegenüber NFS/SMB*
  - *in manchen Situationen auch leichte Nachteile denkbar*
- *sollte bei vergleichbarer Last stärker sein als iSCSI*
  - *erheblich schlankeres Protokoll*
  - *moderneres Design*
- *Performance-Vorteile durch Multithreading*



Performance in der Praxis

Wer viel mißt, mißt ... Vergleiche zu iSCSI / Filesysteme

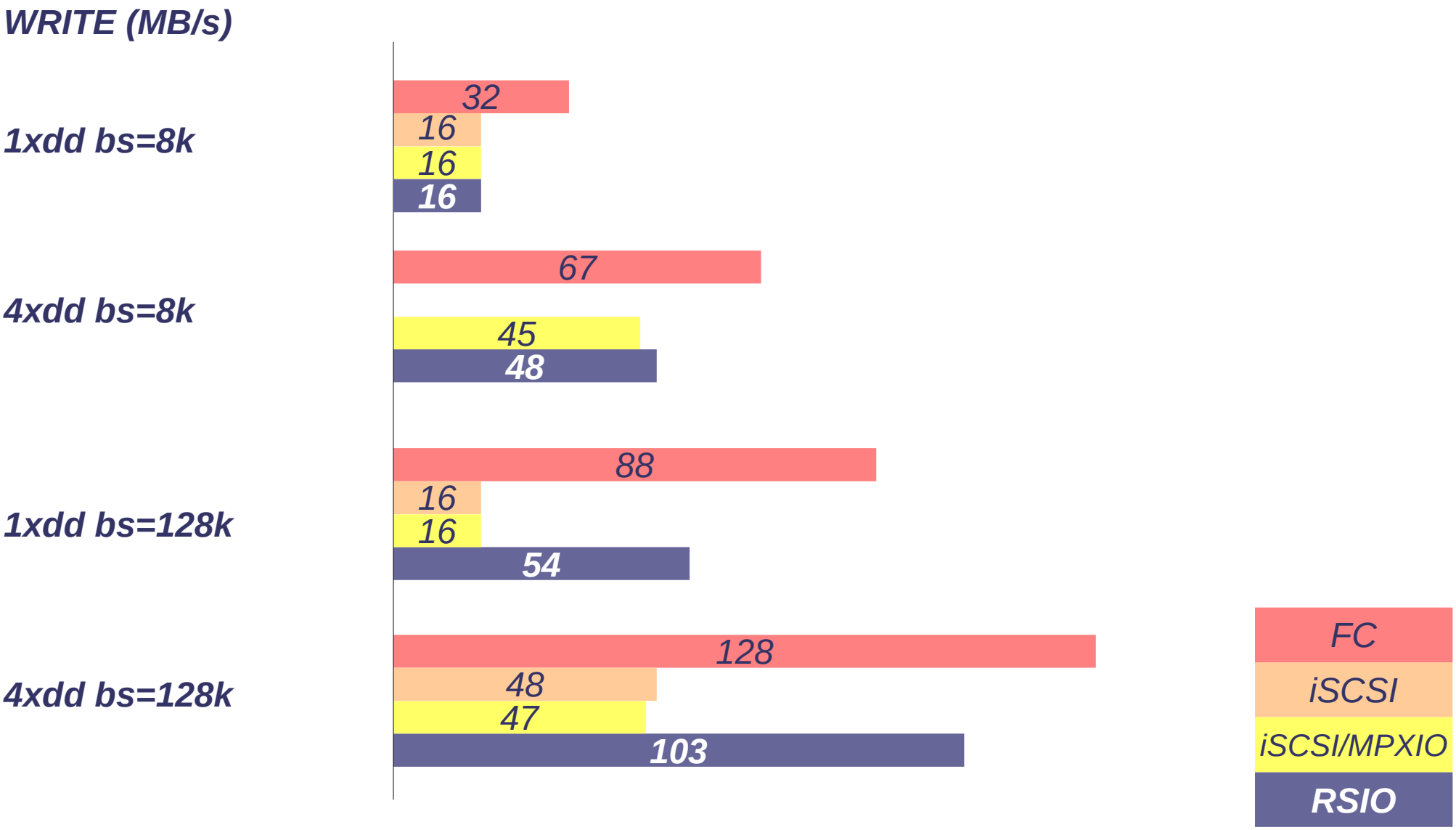


iSCSI

RSIO

Performance in der Praxis

Wer viel mißt, mißt ... Vergleiche zu FC und iSCSI / Character Device (raw IO)



Performance (nicht ganz) in der Praxis

Performance bei Cache-Reads -> der Netzwerkprotokoll-Test

Server-Performance bei Cache Read / 8k

iSCSI	10 Clients	100 Threads	7,6 Cores	**31.000 IOPS**
iSCSI*	10 Clients	100 Threads	10,0 Cores	**85.000 IOPS**
RSIO	4 Clients	64 Threads	5,6 Cores	**98.000 IOPS**
RSIO	4 Clients	128 Threads	6,3 Cores	**102.000 IOPS**

Client-Performance Throughput

RSIO	1 x 1 GBit	< 0,5 Cores	**> 110 MByte/s**
RSIO	2 x 1 GBit	< 1,0 Cores	**> 220 MByte/s**
RSIO	4 x 1 GBit	< 2,0 Cores	**> 440 MByte/s**

Performance und Funktionalität

Details für die Praxis

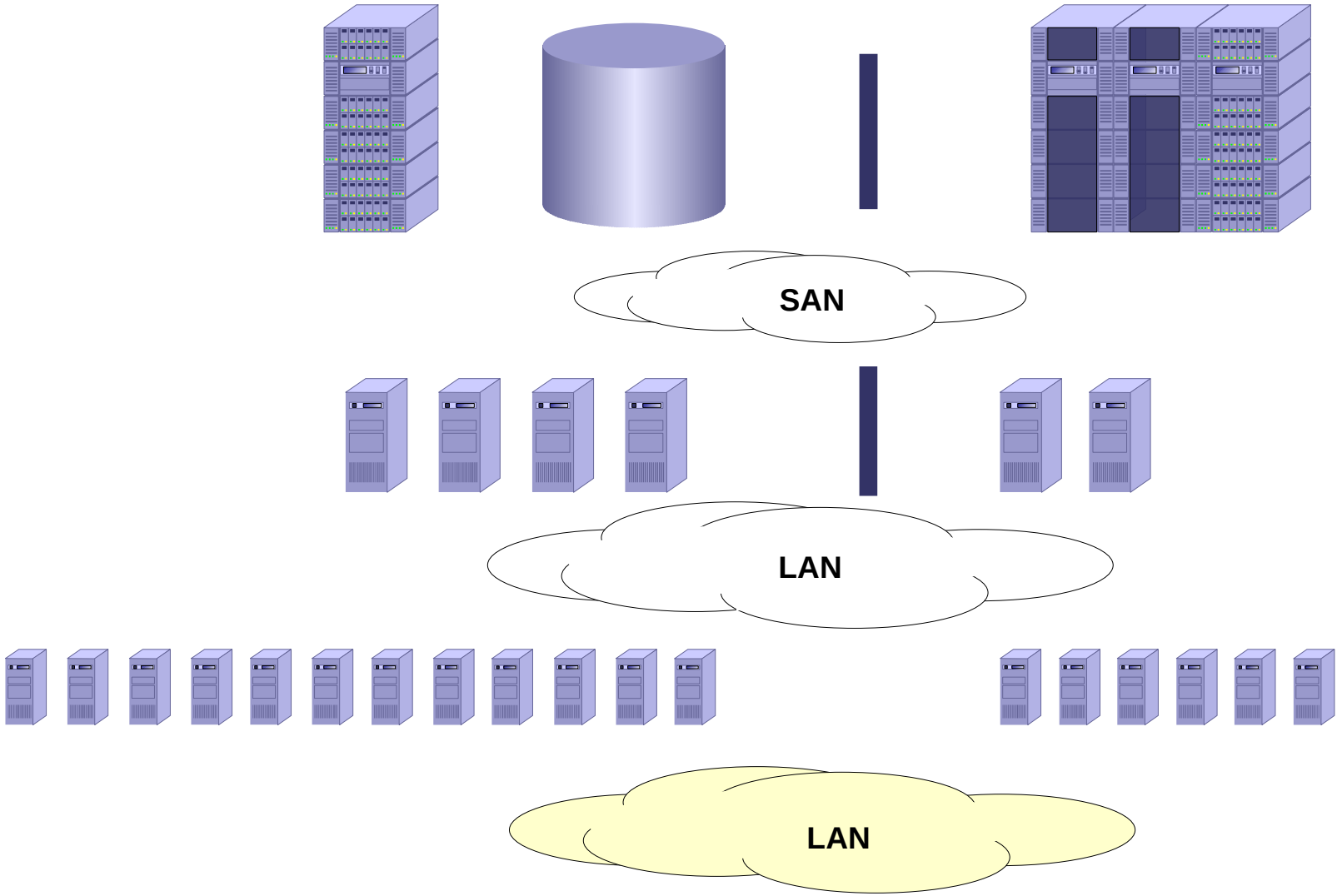
- *exzellente Performance*
  - *keine Spezialsettings für TCP/IP -> Performance “out of the Box”*
  - *bis jetzt TCP, UDP möglich*
  - *prinzipielle Eignung für beliebige Medien*
  - *noch diverse Verbesserungsmöglichkeiten*
- *gewaltige Gestaltungs-, Entwicklungs- und Tuningmöglichkeiten*
- *extrem schlankes Kernelmodul*
- *komplett virtualisierter Speicher darstellbar,*
  Handling wird vom Client aus möglich sein
- *eingebaute Einfachheit (TCP/IP-Handling, Trunking ...)*
- *vielfältige Nutzungsmöglichkeiten / unglaublich viele Szenarien darstellbar*

Wir hoffen auf Ihre Vorschläge und Ideen!

Einsatzszenarien

Was kann ich prinzipiell damit tun?

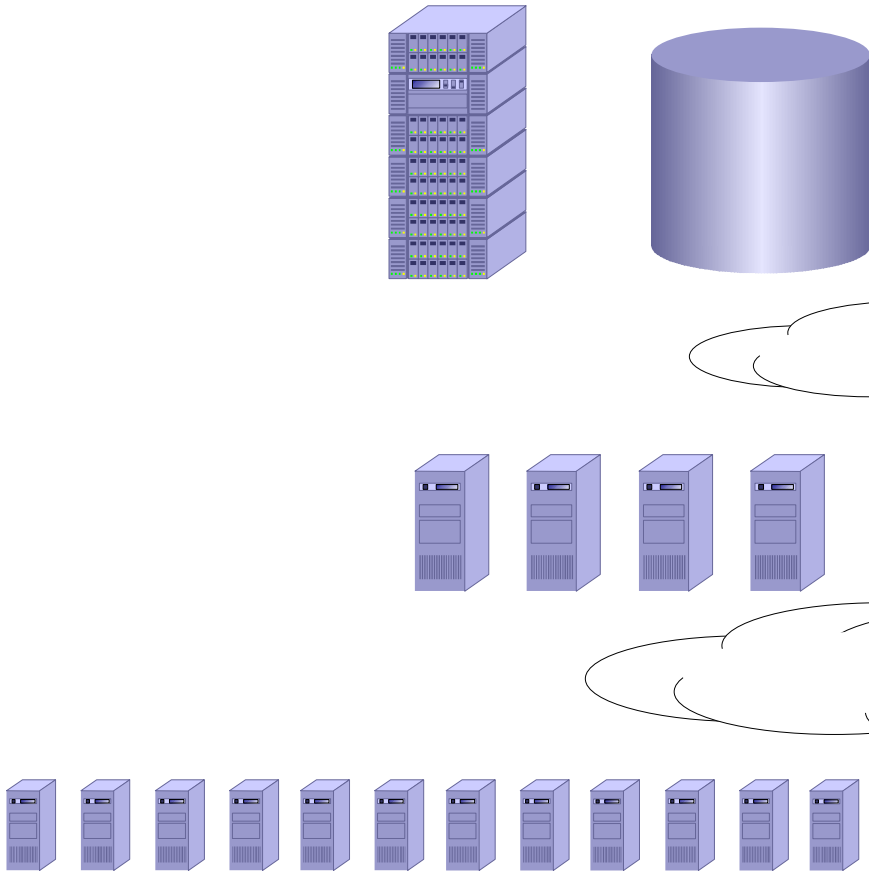
SAN-LAN-Konvergenz



Anwender

Was kann ich prinzipiell damit tun?

SAN-LAN-Konvergenz

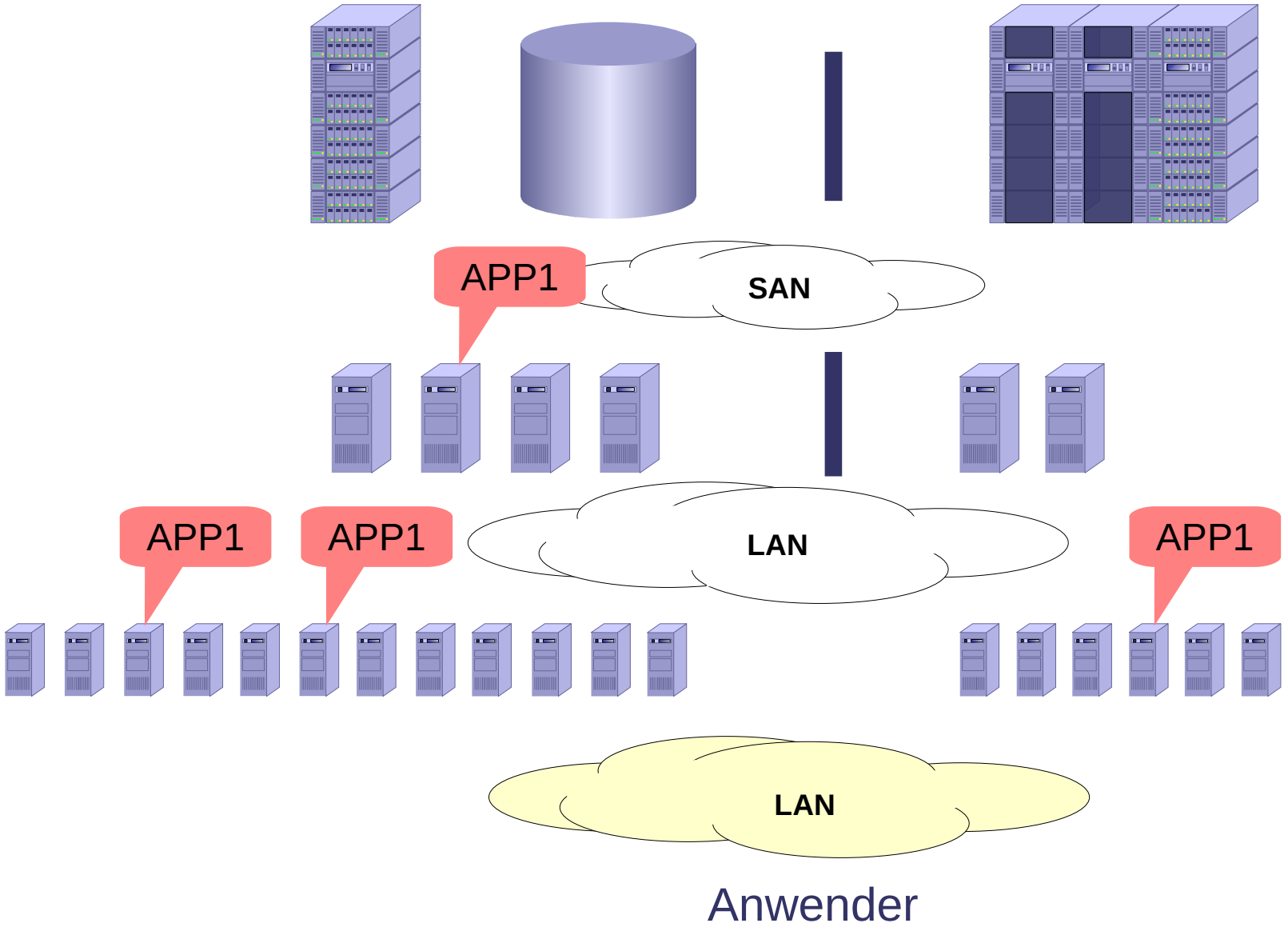


- *Komplexität reduzieren*
- *SAN-Ports einsparen*
- *Portauslastung verbessern*
- *Hosts preiswert und einfach anbinden*
- *Performance-Rightsizing*
- *erleichterte OS-Virtualisierung*
- *Standardkomponenten*

Anwender

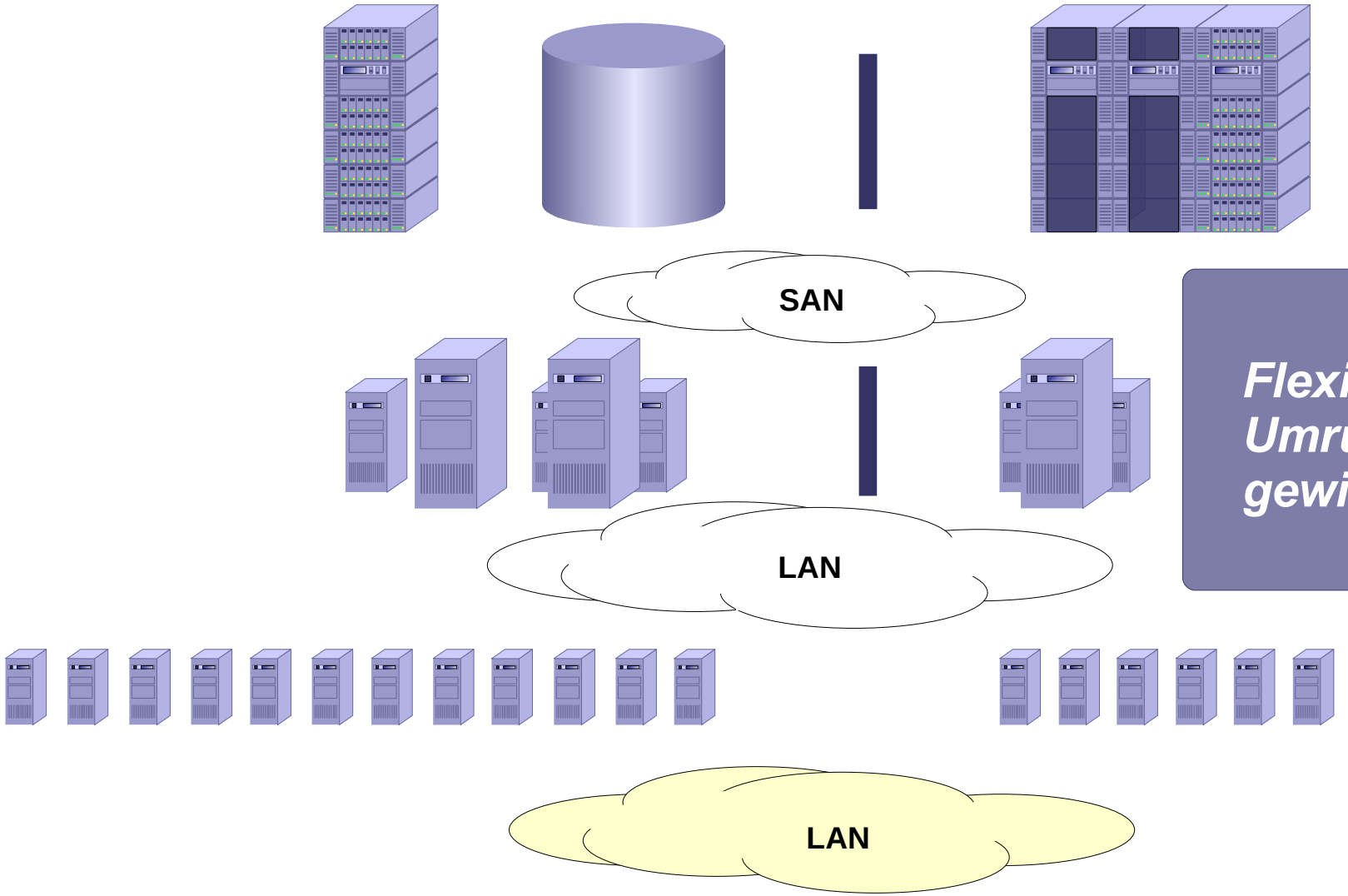
Was wird mit der Hochverfügbarkeit ?

Auch hier: SAN-LAN-Konvergenz



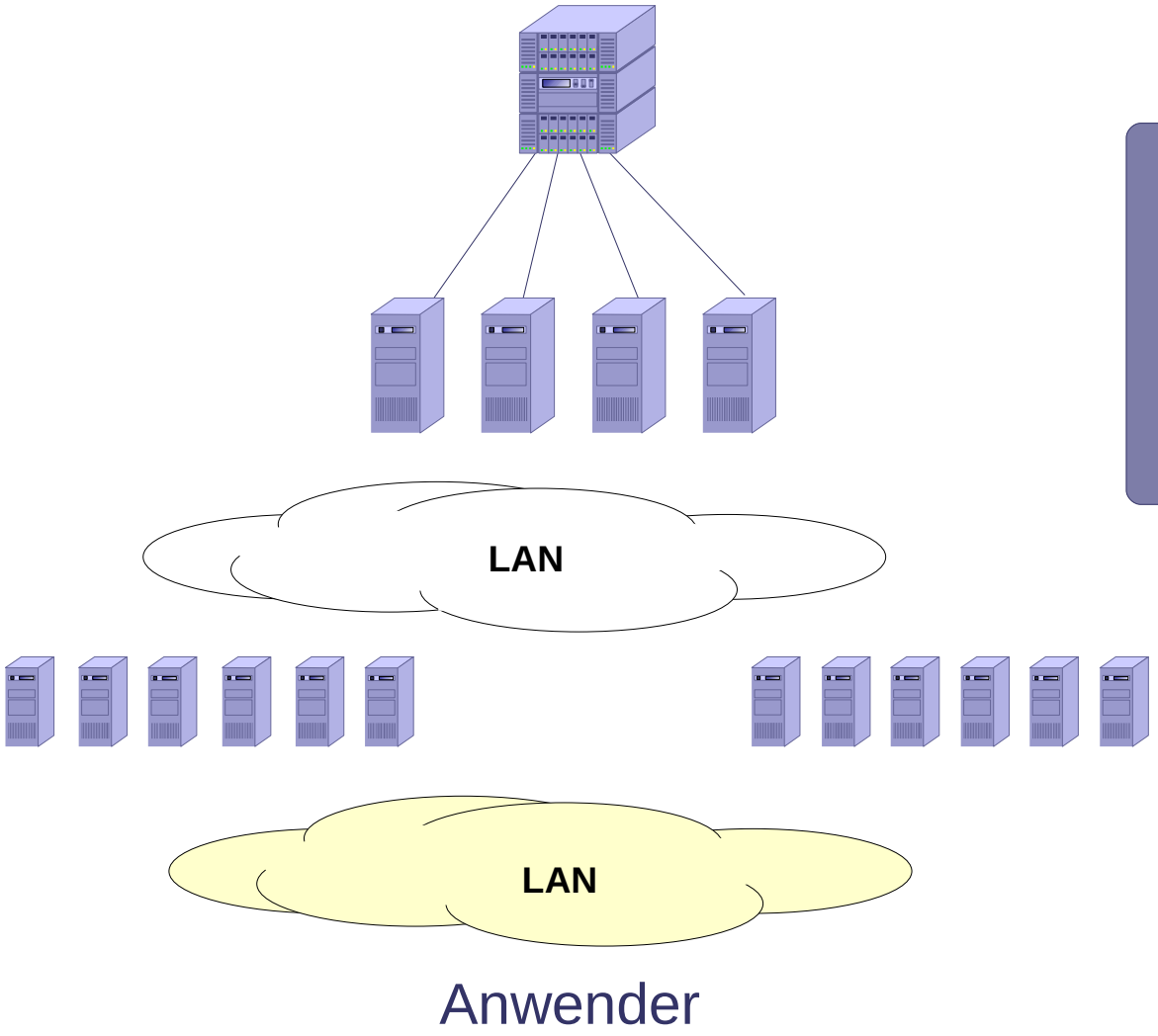
Was kann ich noch anstellen?

SAN-LAN-Konvergenz



Flexibilität bei Umrüstungen gewinnen

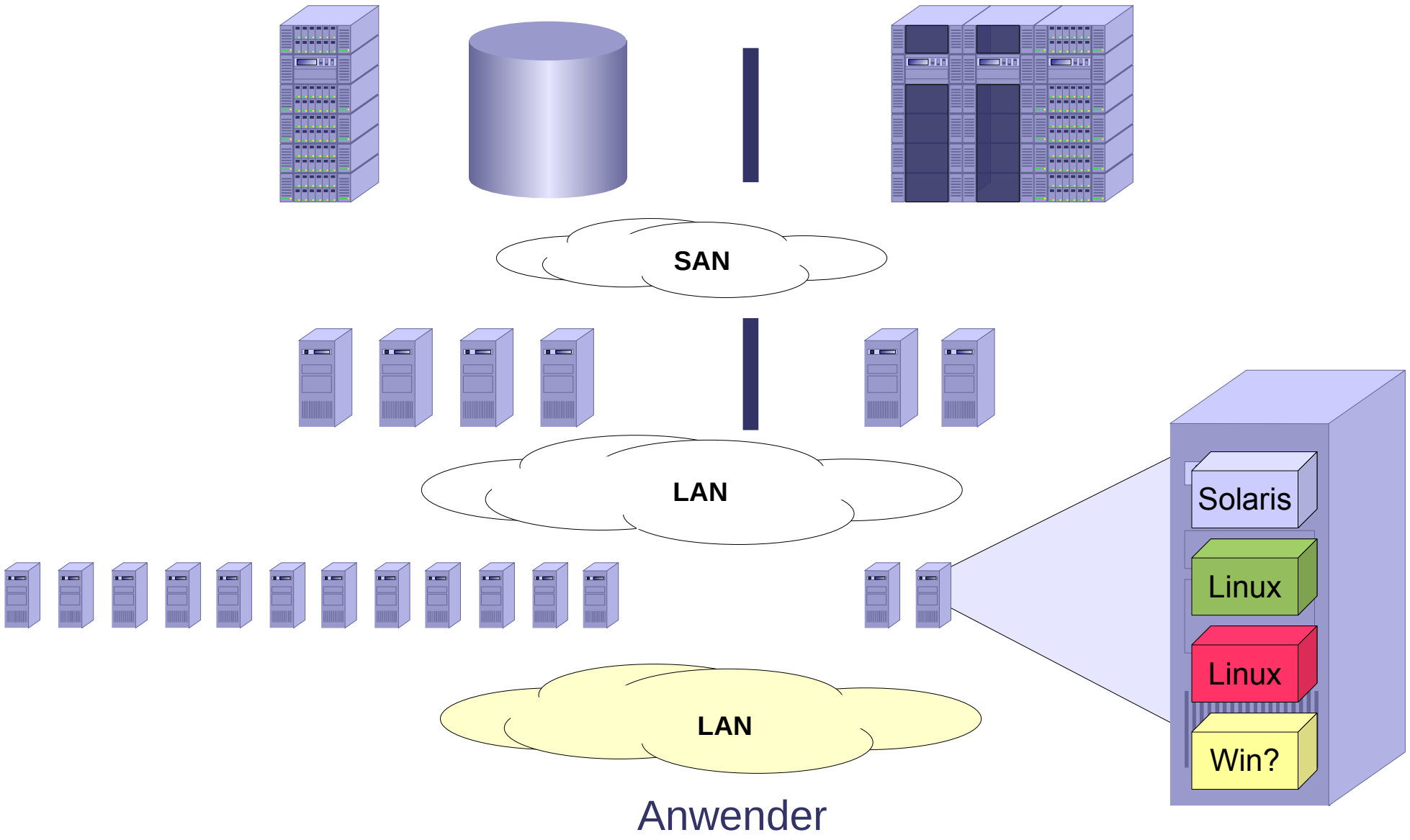
Und auch das geht:
Collapsed SAN = gar kein FC-SAN



ganz auf FC-SAN verzichten, nicht aber auf Funktionalität und Performance

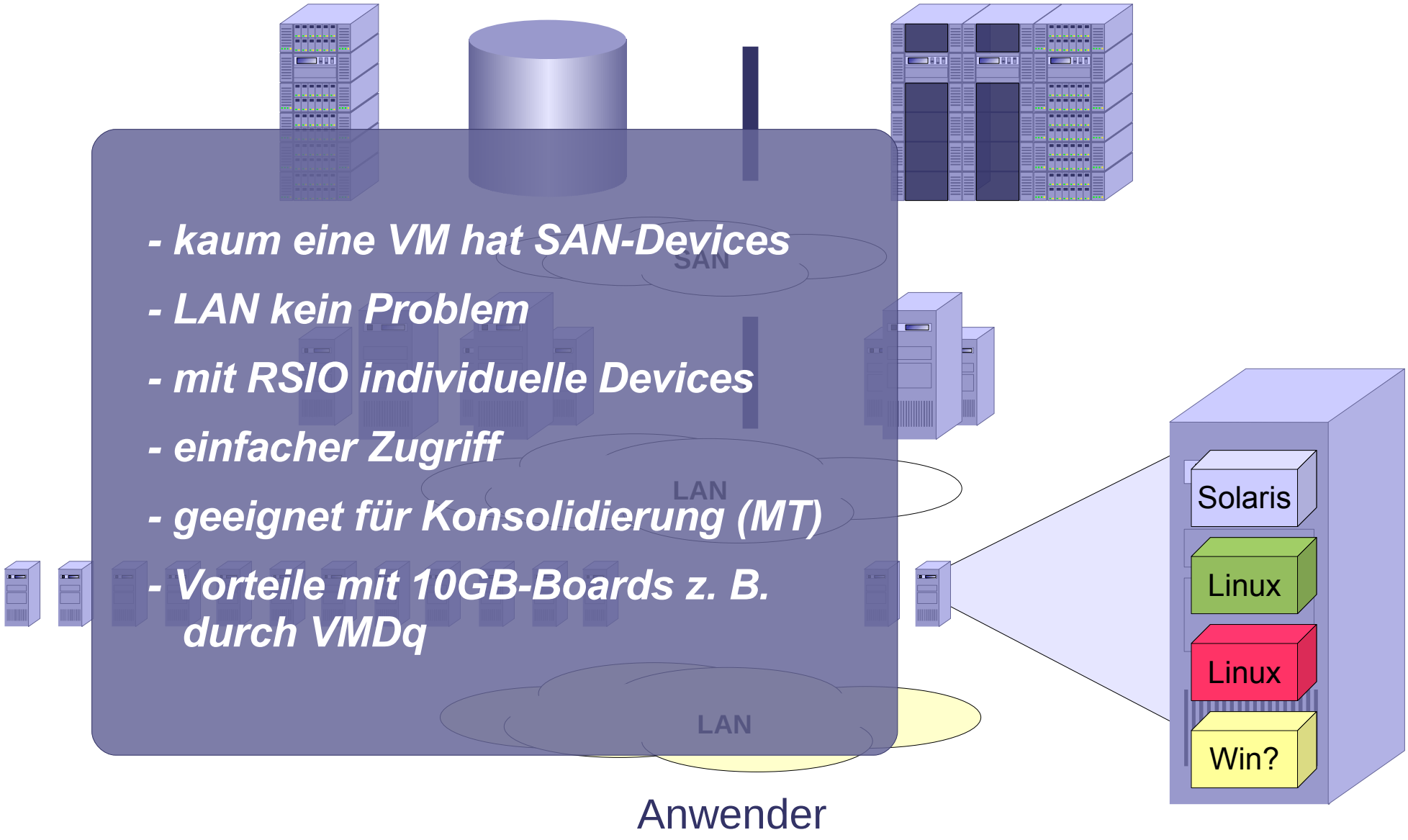
Passend zu Virtuellen Maschinen

Drastische Vereinfachung der Storage-Anbindung für virtuelle Maschinen



Passend zu Virtuellen Maschinen

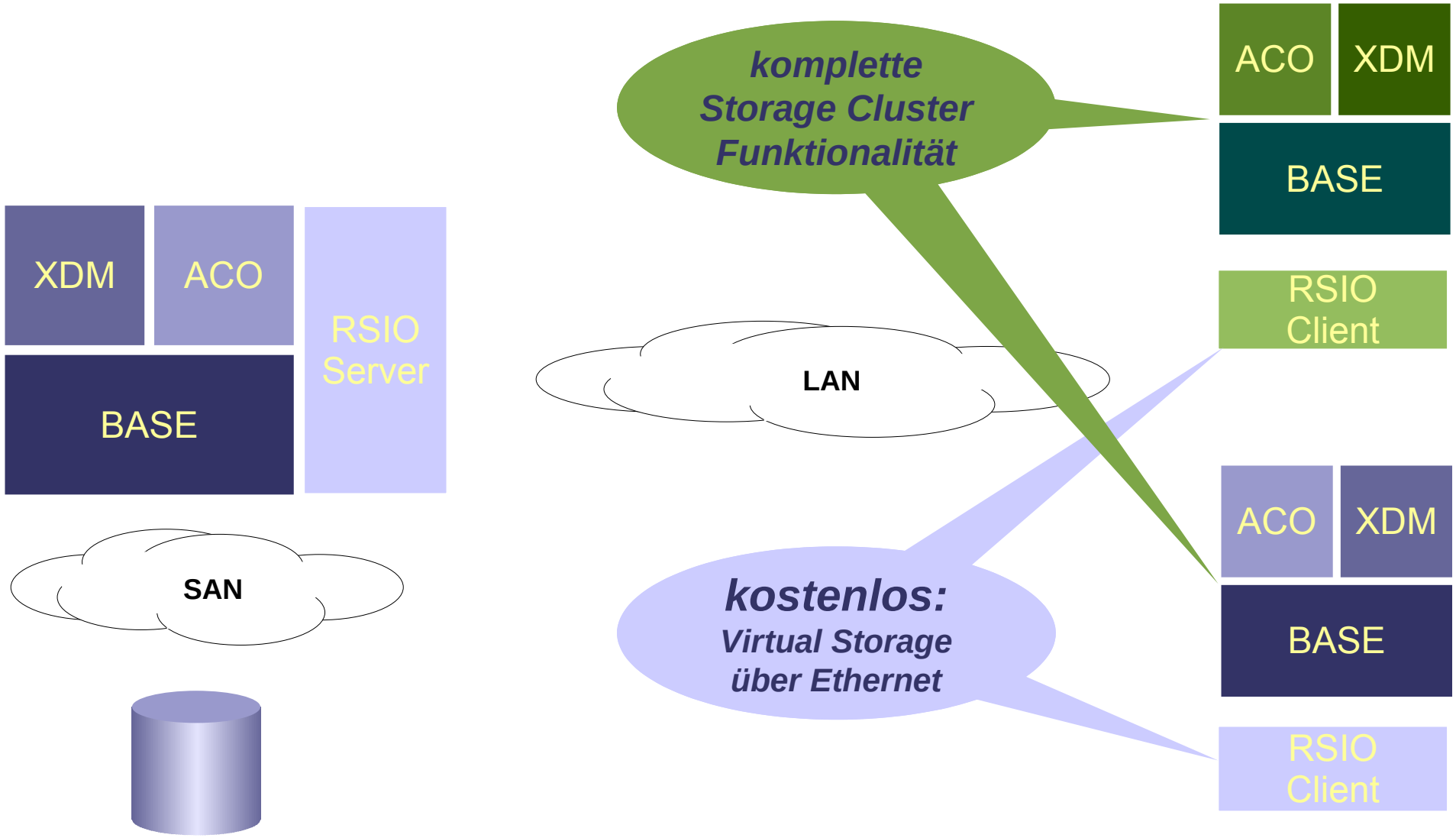
Drastische Vereinfachung der Storage-Anbindung für virtuelle Maschinen



Ausblick

Wie wird es weitergehen?

Aufbruch in neue Welten ...



Die nächsten Schritte

RSIO ist noch immer in einer sehr frühen Phase
Wenn wir uns etwas wünschen dürfen:

- *Testanwender für Solaris + Linux*
- *Feedback und Anregungen*
- *die Möglichkeit, noch vor der Pilotierung Praxiserfahrungen im Design zu berücksichtigen*
- *Vorlauf in der Planung größerer Projekte*

Bitte unterstützen Sie uns und gestalten Sie Ihre Lösung mit !

Zusammenfassung zu RSIO

- ***Storage-Anbindung via LAN** implementiert*
- ***skalierbar und hochverfügbar***
- ***respektable Performance***
- ***einfach** in der Handhabung*
- *unschlagbar **niedrige Kosten***
- *enge Verknüpfung mit Clusterdiensten möglich*
- *ermöglicht virtuellen Storage, virtuelle Ablaufumgebungen und am Ende applikationsbezogenes Management vom Netzwerk-Client aus*
- *geeignet für Virtuelle Maschinen*
- *OSL Storage Cluster erschließt neue Systemplattformen*
- *noch riesiges Entwicklungspotential*

OSL RSIO

Remote Storage I/O

Storage-Networking der nächsten Generation

Besuchen Sie uns noch 2010:

Sun Breakfast	Berlin	25. Juni
OSL Technologietage	Berlin	15./16. September
SNW Europe	Frankfurt/M.	26./27.Oktober
iX Solaris Day	Stuttgart	30.9. - 1.10.

OSL Gesellschaft für
offene Systemlösungen mbH