Welcome to the speech:

Active/Active HA Clustering on Shared Storage with Samba

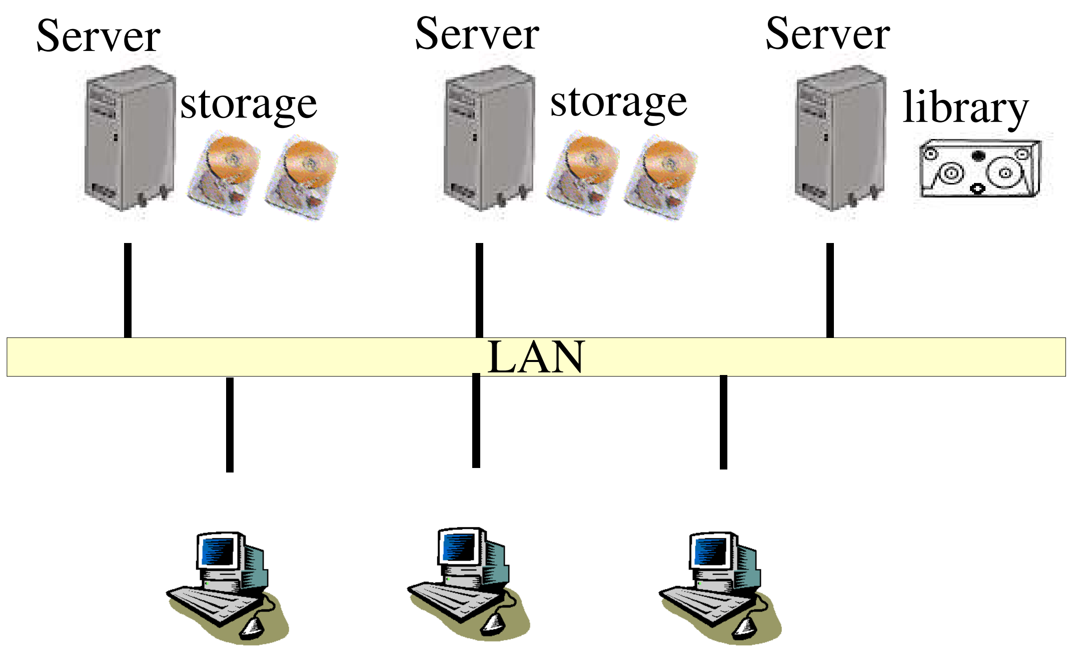
Speaker:	Dipl.-Ing. Thomas Merz / ATIX GmbH
Date:	05/04/2005 – SambaXP 2005
eMail:	merz@atix.de

ATIX

Outline

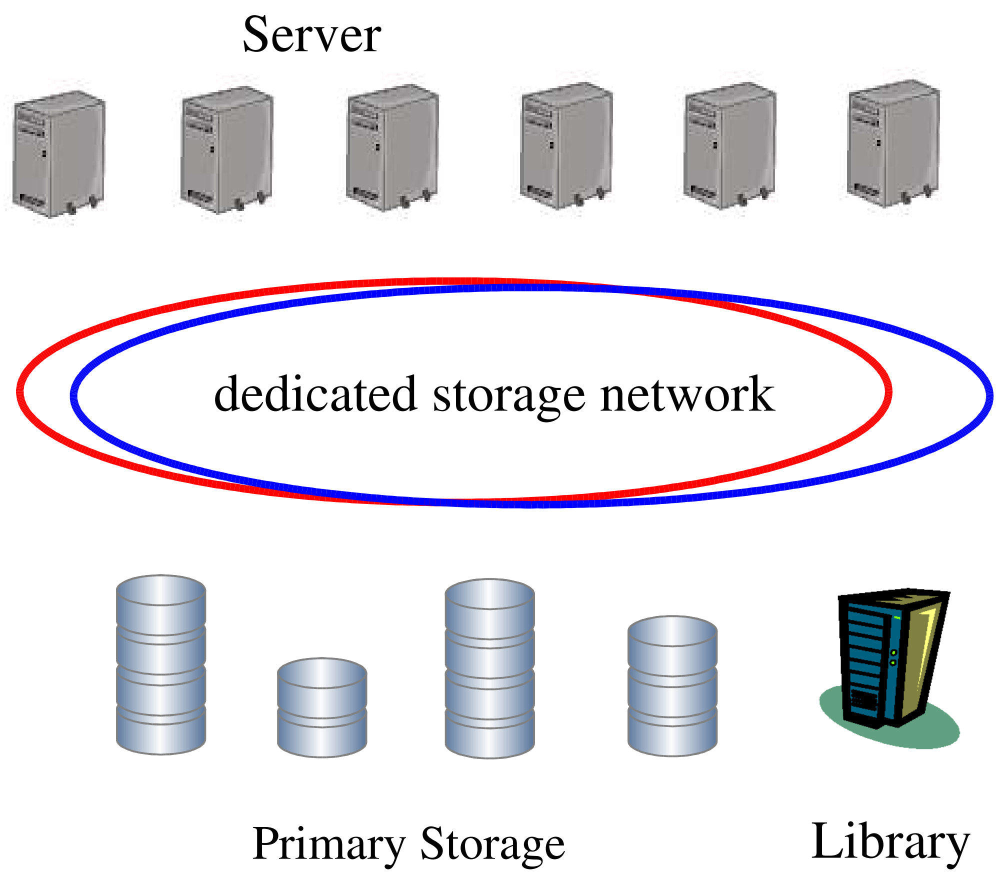
- Infrastructure Types & Storage Virtualization
- HA Clustering
  - Types and Possibilities
  - Application Problems
  - Filesystem Issues
  - Shared Storage
- Samba and Shared Storage
- Outlook

„Classical“ Infrastructure



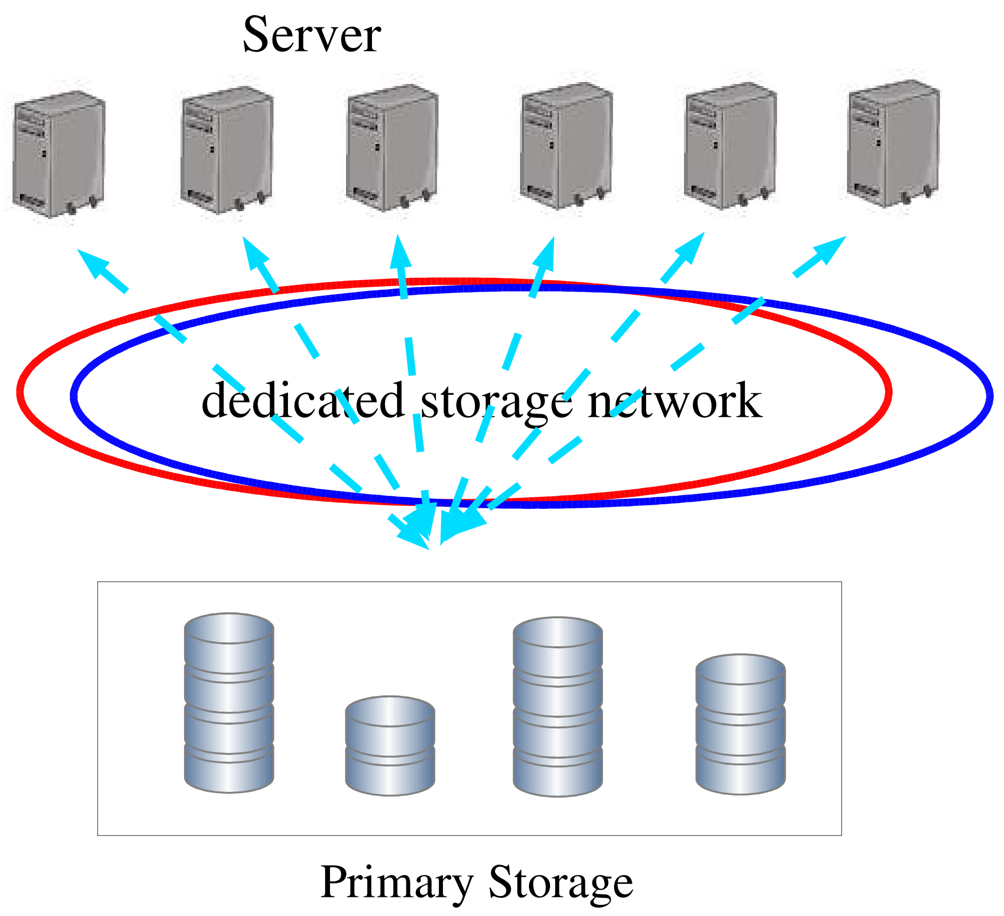
- DAS (Direct Attached Storage)
- Classical: Parallel SCSI
- Server and Storage are one unit
- bad catastrophe precautions possible
- complete communication using the very same LAN
- Performance and scaling bottleneck

Storage Network



- Server and storage connection using a dedicated storage network
- Spacial separation of servers and storage possible
- Servers are just „sheet“
- Flexible assignation of server <-> storage
- Performance and availability become independent from the LAN

Storage Network



- Servers access the physically identical storage
- to be used for:
  - HA Cluster
  - High Performance Cluster
  - Storage Cluster
  - Single system configuration („shared root“)

Storage Virtualization

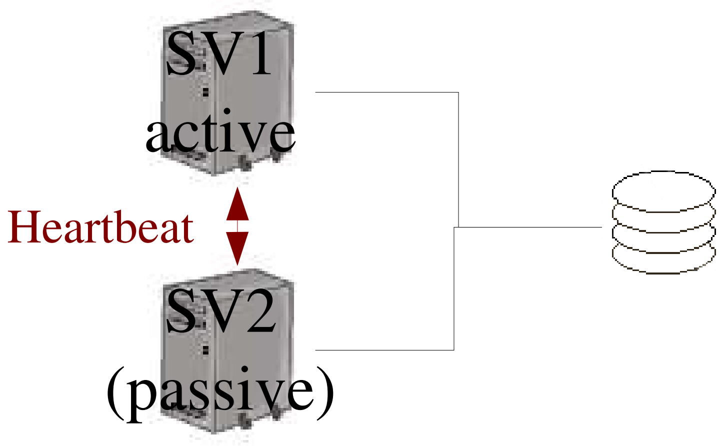
- Layer to separate the host- from the storage-layer
- The physical storage architecture becomes „invisible“ to the hosts
- Transparency of the data's physical location
- Dynamic infrastructure
  - Transparent „on the fly“ reconfiguration
  - Transparent movement and reorganization of data (HSM)
- Many different levels of storage virtualization possible
- Software- and hardware components are part of the virtualization

Storage Virtualization

The filesystem level is part of the storage virtualization

- If a „cluster file system“ is mentioned on the upcoming slides GFS (RedHat's Global File System) is the one being used
  - All nodes share the management of the cluster fs
  - No dedicated Fileserver (SPoF) needed
  - FS consistency done using a special cluster wide locking service
  - Best used in SAN based infrastructures
  - Standard applications might need design review to work properly on a shared system like this
  - Diskless (pure SAN based) shared root operation possible

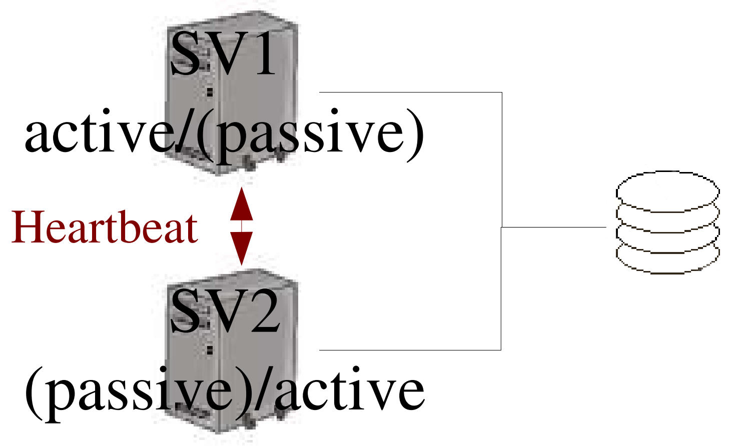
HA Cluster Active/Passive



Concept:

- The passive node waits for the active one to fail
- One or more services are running on the active node
- A software component is responsible for monitoring the services' and servers' health
- Different implementations possible
- A single host filesystem has to be transferred from one node to the other in case of failover

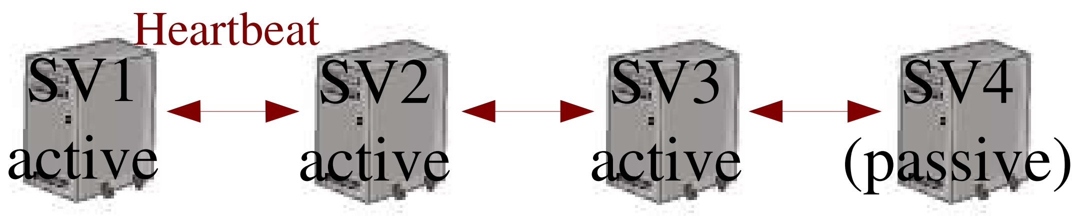
HA Cluster Active/Active



Concept:

- Different services run on each of the nodes
- Each node is active and passive at the same time
- In case of a failover performance will be lower on the „double active“ node
- In case of single host filesystems each service needs its own filesystem (or multiple services have to failover together, e.g. NFS & Samba)

HA Cluster N+1



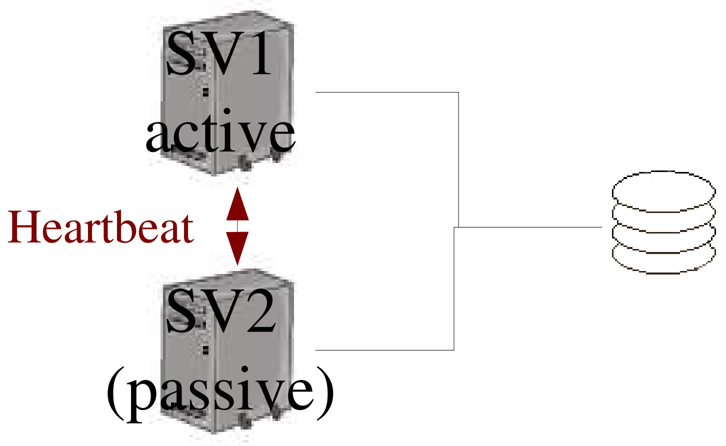
Concept:

- More servers than necessary are used (N+1, N+2, ...)
- Cluster can act without performance breakdown in case of a failover
- Failover softwares normally have a limited number of supported nodes
- Filesystem(s) have to be transferred from node to node in case of a failover
- Same rules and limitations apply as in the standard active/active & active/passive scenario

Stateless and Stateful Services

- Problem of <u>transparent</u> failovers
- Theoretically a service has to be restarted on the failover node in the same state as it „left“ the failed node
- *Stateless* service do not rely on memory based states
  - => Transparent failover is no (not a huge) problem
- *Stateful* services have saved states in the local nodes' memory. They have to be saved and transferred to the other node to prevent data loss.
  - => Tranparent failover is a (huge) problem
  - => An open connection cannot be re-established
    Data loss occurs

Filesystem Issues



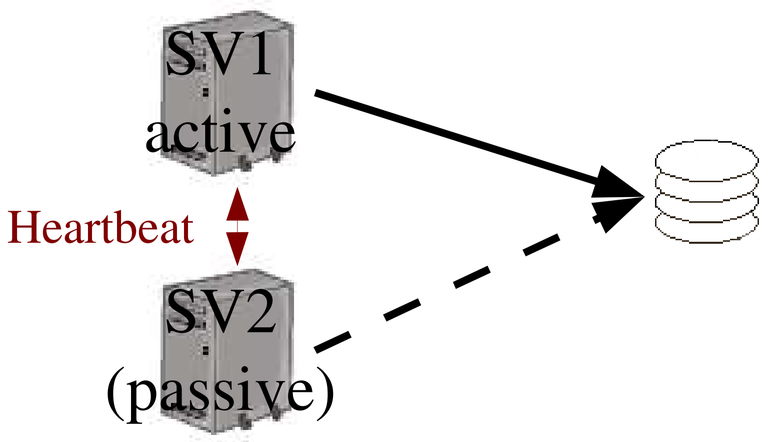
Problem: Exclusive or parallel data access

A HA cluster must grant access to the same data to two or more nodes without corrupting the data

Possibilities:

- Data replication between two or more volumes
- Standard single host filesystem is mounted „ping-pong“
- Clusterfilesystem is permanently mounted on all hosts

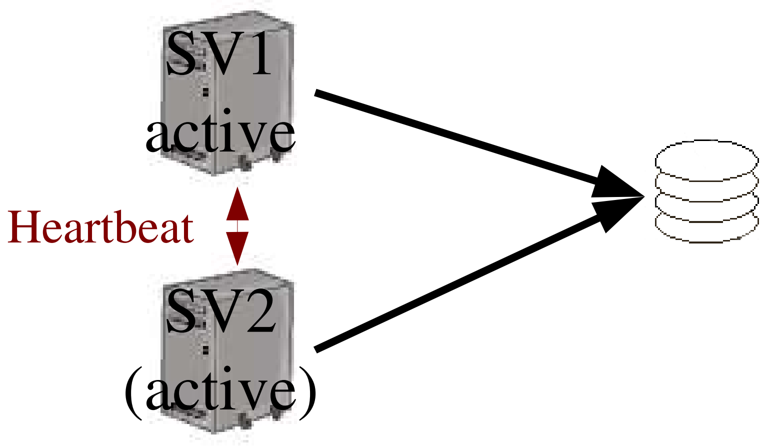
Filesystem Issues



Standard Single Host Filesystem „ping-pong“ mount

- Shared Storage (SAN, Shared SCSI Bus) necessary
- SAN based solution expensive, Shared SCSI-Bus limited to two hosts
- Each host can act as active host for one application (= one filesystem) and act as passive host for an other application (= other filesystem)
- Multiple filesystems per host possible
- Filesystems must be transferred in case of a failover
- Local file access can interrupt a filesystem failover

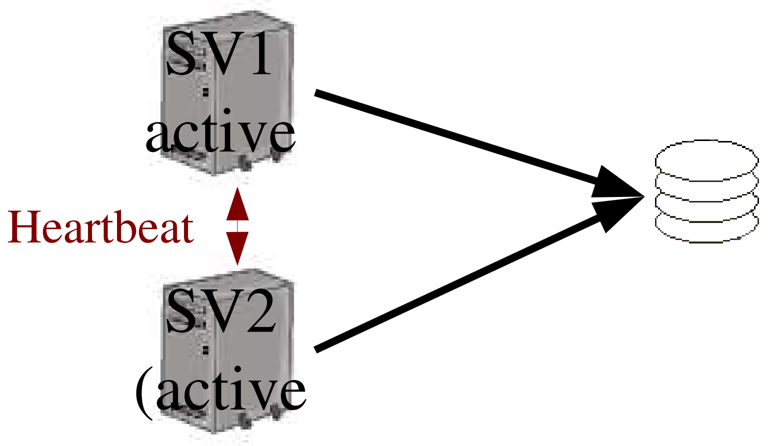
Filesystem Issues



Clusterfilesystem is permanently mounted on all hosts

- Shared Storage (SAN) necessary
- SAN based solution is expensive, number of cluster nodes „not“ limited
- Each node can act as active host for a application with a shared filesystem among all nodes
- Depending on the application's design „concurrent“ write access is possible to the physical same file
- Most applications support concurrent read access to the physical same file
- Active Loadbalancing is possible using every cluster node

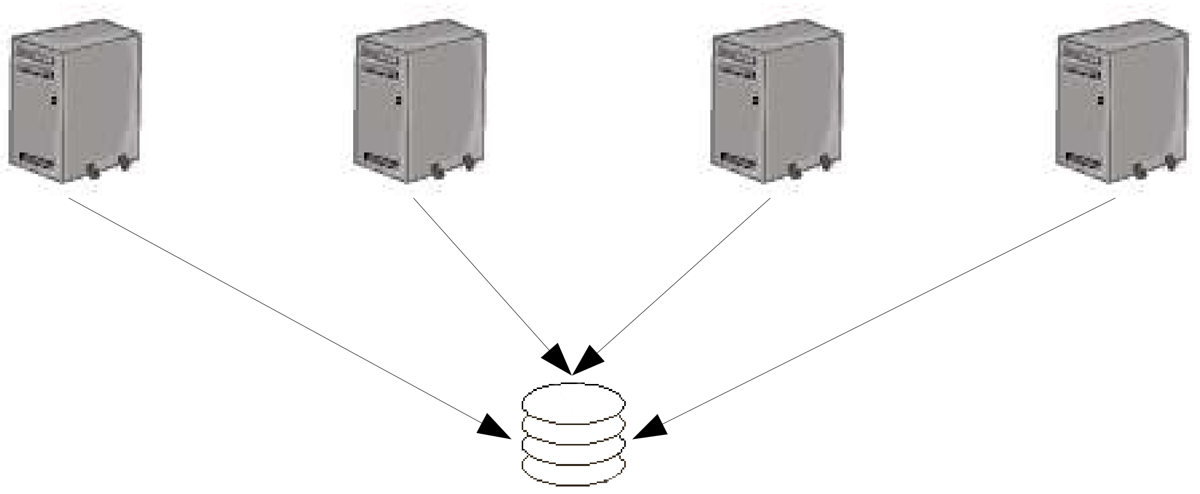
Filesystem Issues



Clusterfilesystem is mounted permanently on all nodes

- All hosts share the management of the file system among each other
- No dedicated fileserver (eg.: NFS) necessary
- No Single Point of Failure (SPoF) if a redundant infrastructur and redundant lock servers exist
- Filesystem access directly in the SAN, no detour via IP based connections

Clusterfilesystem Issues



- All hosts see the physically identical data and have full write and read access to them
- Access restrictions are done *only* in the application layer
- Each cluster node can take over the role of any fellow node
- Each node can run any service (NFS, Samba, Mail, Web, ...)
- Number of cluster nodes „not“ limited
- Very fast failover scenarios possible (no remounts, no service stops and starts, ...)

HA on shared storage

Example: Samba fileservices

Different HA mechanisms need different configurations

- HP ServiceGuard
  One smbd service runs per node, all shares are served using this service
  - One *smb.conf* per node with one *[global]* section
  - All share configurations are done within this file
  - One nmbd service per node, running failover service independent
  - One winbindd service per node, running failover service independent
  - A Samba failover just adds the „new“ share configurations to the existing *smb.conf* and reloads the service

HA on shared storage

Example: Samba fileservices

Different HA mechanisms need different configurations

- RedHat Cluster Service
  Multiple smbd services run per node, shares can be assigned to the appropriate one. Samba services are independent from each other.
  - One *smb.conf* per service with one *[global]* section
  - All share configurations for this service are done within this file
  - One nmbd service per smbd configuration, running failover service dependent
  - One winbindd service per smbd configuration, running failover service dependent
  - A Samba failover „moves“ a complete samba configuration (smbd, nmbd, winbindd, smb.conf-File[s]) to the „new“ node

HA on shared storage

Example: Samba fileservices

Different HA mechanisms need different configurations

- HP's ServiceGuard
  - Only one failover service necessary and possible for Samba
  - Various script patches necessary to get Samba running
  - Winbindd and nmbd services are failover independent

- RedHat Cluster Service
  - More flexible configurations possible
  - Independent Samba services per node possible
  - Easy adding and removing of shares to a service
  - Script code patch necessary for winbindd and nmbd
  - Winbindd and nmbd are failover dependent

Samba and shared storage

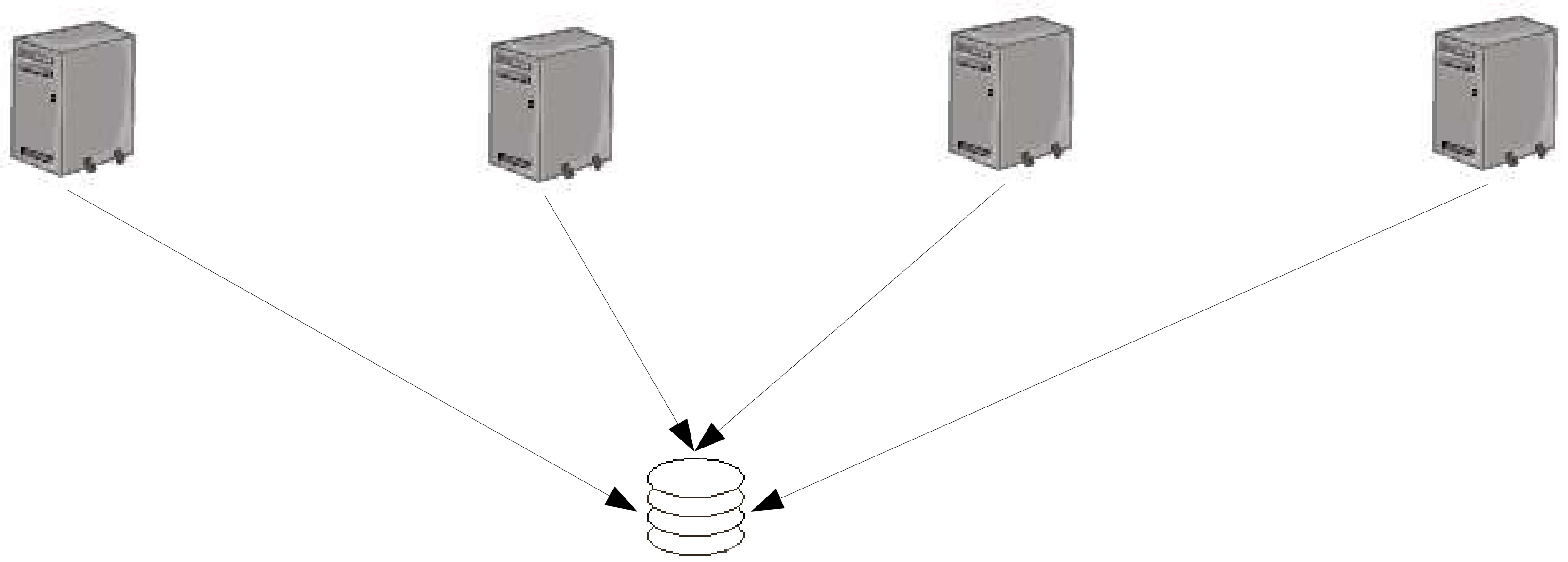
Example: Samba fileservices

[share1],[share2]
192.168.10.50&51
server1

[share3]
192.168.10.60
server2

[share4]
192.168.10.70
server3

[share5],[share6]
192.168.10.80&81
server4



Common Filesystem: /mnt/gfs/samba/ on SAN Storage

Samba and shared storage

Example: Samba fileservices

Structur:

Share 1:	/mnt/gfs/samba/shares/share1/
Share 2:	/mnt/gfs/samba/shares/share2/
Share 3:	/mnt/gfs/samba/shares/share3/
Share 4:	/mnt/gfs/samba/shares/share4/
Share 5:	/mnt/gfs/samba/shares/share5/
Internal:	/mnt/gfs/samba/internal/

„Internal“ keeps necessary parts of the samba configuration which have to be shareable among all cluster nodes (e.g. smb.conf versions).

Clusterfilesystem Issues

Example: Samba fileservices

Fragmentation of the „Internal"-Directory:

```
ls -la /mnt/gfs/samba/internal/ (selective)
        lrwxrwxrwx    root    root    /global -> @hostname
        drwxr-xr-x    root    root    /server1
        drwxr-xr-x    root    root    /server2
        drwxr-xr-x    root    root    /server3
        drwxr-xr-x    root    root    /server4
        drwxr-xr-x    root    root    /include
```

Server1 accesses */mnt/gfs/samba/intern/global/* logically,the physical location is */mnt/gfs/samba/intern/server1/.*

Server2 and all other servers would physically access the directories *server2, server3, ...* This is called *Context Dependant Path Name* (CDPN)

Parts of the samba configuration which have to be unique for the different nodes can be placed here without the need to adapt path names whenever (e.g. failover) a configuration has to be changed.

Samba and shared storage

Example: Samba fileservices

- The subdirectory *internal* contains share information for all shares

  e.g.: *share1.conf*

```
[share1]
path        =   /mnt/gfs/samba/shares/share1/
comment     =   Share1
read only   =   no
...
```

  e.g.: *share2.conf*

```
[share2]
path        =   /mnt/gfs/samba/shares/share2/
comment     =   Share2
read only   =   yes
...
```

Samba and shared storage

Example: Samba fileservices

- The subdirectory *server1* contains the config file *smb.conf* for *server1*

  e.g.: *smb.conf* (accessible for *server1* via */mnt/gfs/samba/global/*)

```
[global]
...
...
...

include                 =    /mnt/gfs/samba/internal/include/share1.conf
include                 =    /mnt/gfs/samba/internal/include/share2.conf
```

Samba and shared storage

Example: Samba fileservices

- The subdirectory *server2* contains the config file *smb.conf* for *server2*

  e.g.: *smb.conf* (accessible for *server2* via */mnt/gfs/samba/global/*)

```
[global]
netbios name        =   %h
encrypt passwords   =   yes
security            =   ads
...
include             =   /mnt/gfs/samba/internal/include/share3.conf
include             =   /mnt/gfs/samba/internal/include/share2.conf
```

- *share2.conf* is now served using two samba servers in parallel. This is only possible on read only shares or if the same physical files are not accessed in parallel for write access. Otherwise data corruption *will* happen. No reliable locking is possible between two independent samba services if the accessing application has no appropriate locking mechanism.

Samba and shared storage

Example: Samba fileservices

Loadbalancing:

- No load balancing possible in Active/Passive configurations
- In an Active/Active environment load balancing can be done manually by assigning the shares to different nodes
- Active/Active configurations can be limited to serve different file locations per active service if single host file systems are used
- Active/Active configurations can be used to concurrently access the same filesystem. Precautions have to be taken to avoid concurrent write access to the very same file at the same time. Load balancing is possible in giving multiple Samba servers read access to a share and use an other one for writing. Load balancing within a client connection is not possible.

Other suitable applications

- Similar solutions are possible with services like NFS as well, NFS is bit more tricky to implement due to different file handling
- NFS- and Samba mixed mode environments are possible
- Other possible services for failover use:
    - Apache
    - Sendmail/Postfix
    - Courier IMAP
    - MySQL
    - ...

ATIX

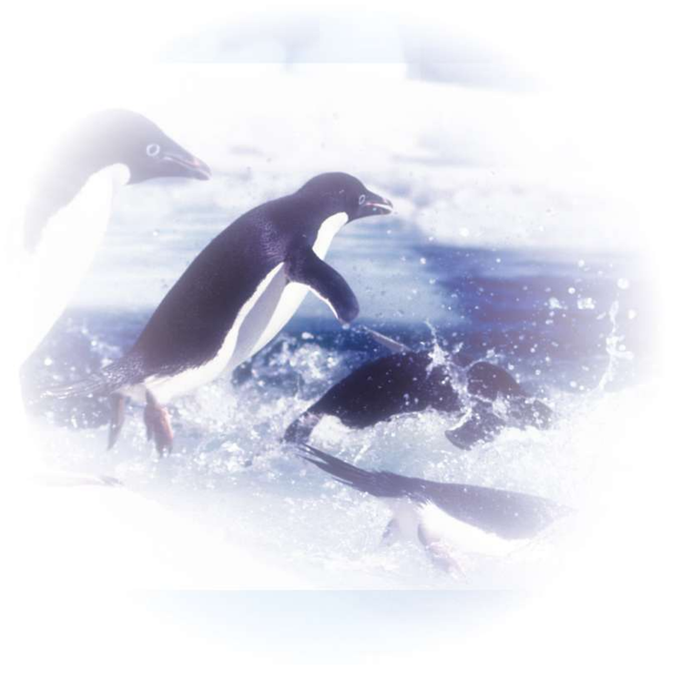
Outlook

Shared Root Cluster

- Examples mentioned before assumed a local OS on each host
- In a shared root cluster every node can be booted directly from the SAN storage, local harddisks are not necessary (except from swap space).
- Every node is identical to its fellow hosts (reduction to pure CPU power)
- Changes/Updates/... can be made at one central point and are available throughout the whole cluster immediately
- For more information see iX 11/2004 (page 118) „Aus dem Pool schöpfen“ or online:
  - http://www.redhat.com/magazine/006apr05/features/gfs/ (english version)
  - http://www.redhat.de/pdf/0410iXMagazinRed_Hat-Atix-Artikel.pdf (german version)

Thank you!

Any Questions?



Atix GmbH
Einsteinstr. 10
87516 Unterschleißheim
www.atix.de
info@atix.de