

The Unbreakable Database System

Real Application Cluster

Unterführung, 04.2005

M. Kühn

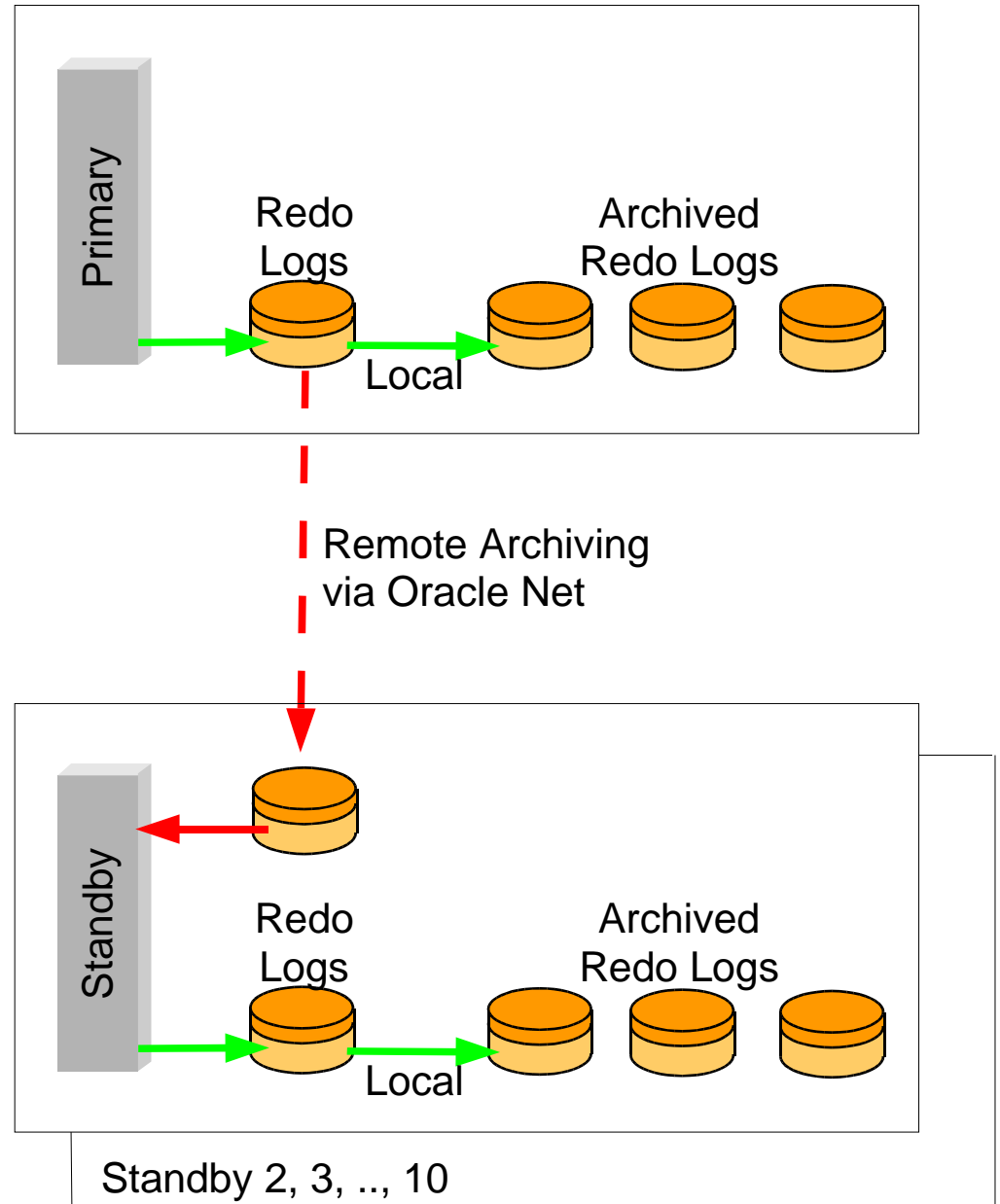
- **Comparissoon HA**
 - HA Ziele, DataGuard, HA Oracle, RAC
- **RAC Features**
 - Cache Fusion, TAF, Load Balancing
- **RAC on Solaris**
 - SPARC, x86
- **RAC on Linux**
- **Konfiguration**
 - „best Practice“ - Scaling

Comparisson HA: HA Ziele

- Reduzieren bzw. Vermeiden von Ausfallzeiten
 - Software- , Hardware- , Human Error, Disaster
- Daten und Anwendungen für Benutzer verfügbar halten
- Erhöhung des Applikationsdurchsatzes durch horizontale Skalierung
- Erhöhte Verfügbarkeit auch während Wartung oder Upgrade

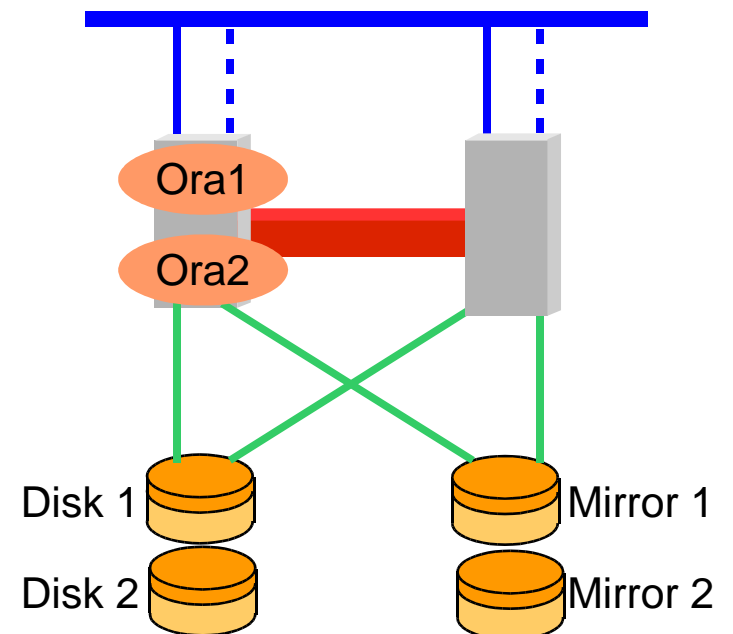
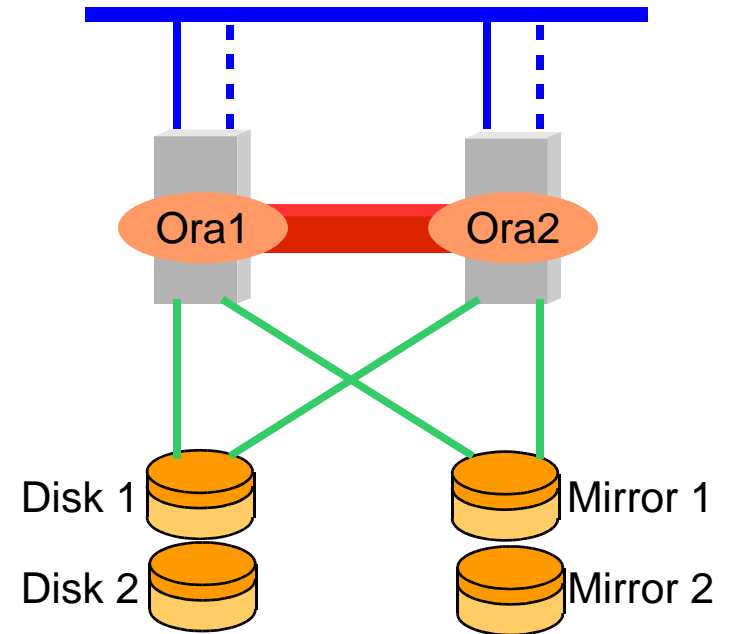
Comparisson HA: DataGuard

- Vorteile
 - Schutz gegen Human Errors
 - Standorte können beliebig sein (Disaster Recovery)
 - Bis zu 10 Standby Server
 - No Data Loss im Guaranteed Protect Mode
 - Gap Detection / Gap Resolution
- Nachteile
 - Kein automatisches Failover
 - Hoher administrativer Aufwand
 - Gracefull Database Failover ist zeitintensiv
 - Datenverlust, wenn nicht im Guaranteed Protect Mode
 - Je höher der Datenschutz, desto schlechter die Performance



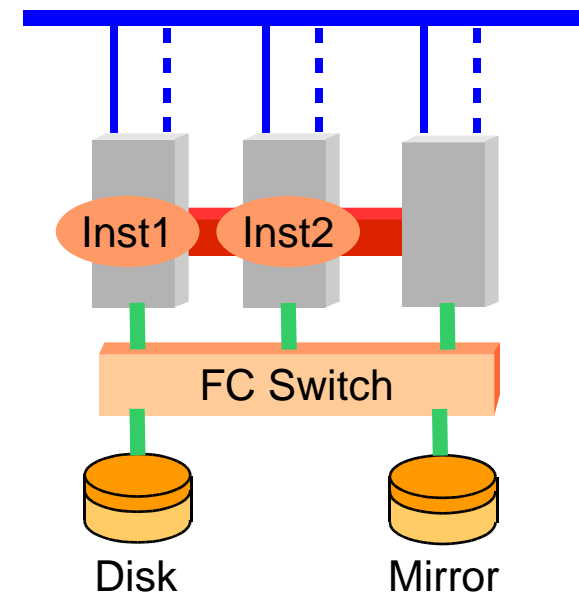
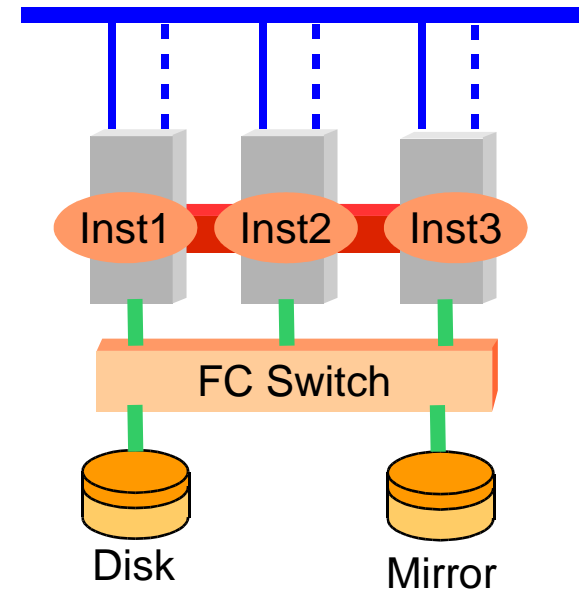
Comparisson HA: HA Oracle

- Vorteile
 - automatische Übernahme im Fehlerfall
 - schnelle Fehlererkennung
 - kein Datenverlust
 - Aktiv / Aktiv Konfiguration
 - geringer administrativer Aufwand
- Nachteile
 - keine horizontale Skalierung
 - Im Fehlerfall müssen sich alle User wieder verbinden
 - Standort maximal 10km getrennt (Campus Cluster)



Comparisson HA: Oracle RAC

- Vorteile
 - Skaliert über alle Knoten
 - Hohe Performance durch Cache Fusion
 - Im Fehlerfall sind nur wenige Benutzer (hier 33%) betroffen
 - Automatische Benutzerübernahme
 - Sehr schnelle Übernahme (<1min)
 - Applikationen werden fortgeführt (TAF)
- Nachteile
 - Hohe Kosten
 - Standort maximal 10km getrennt



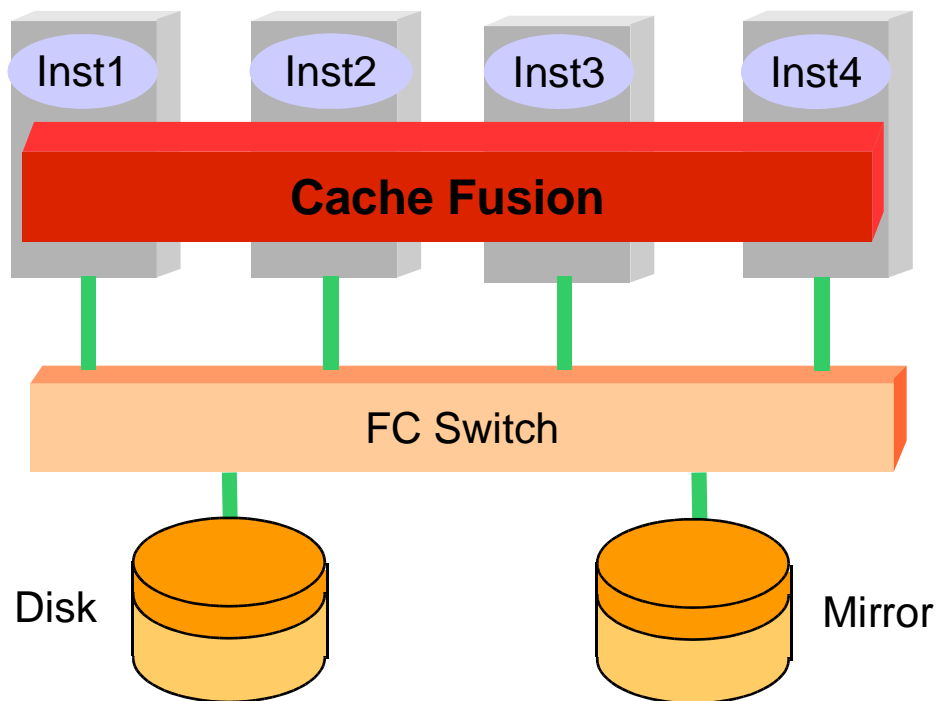
Comparisson HA: Bewertungstabelle

<u>Criteria</u>	Data Guard	HA Oracle	RAC
Performance	Low	Normal	High
Availability	Normal	High	Very high
Failover	8-10min	2-3min	< 1min
Data Loss	High	Low	Very low
Manageability	Complex	Easy	Easy
Cost	Low	Normal	High

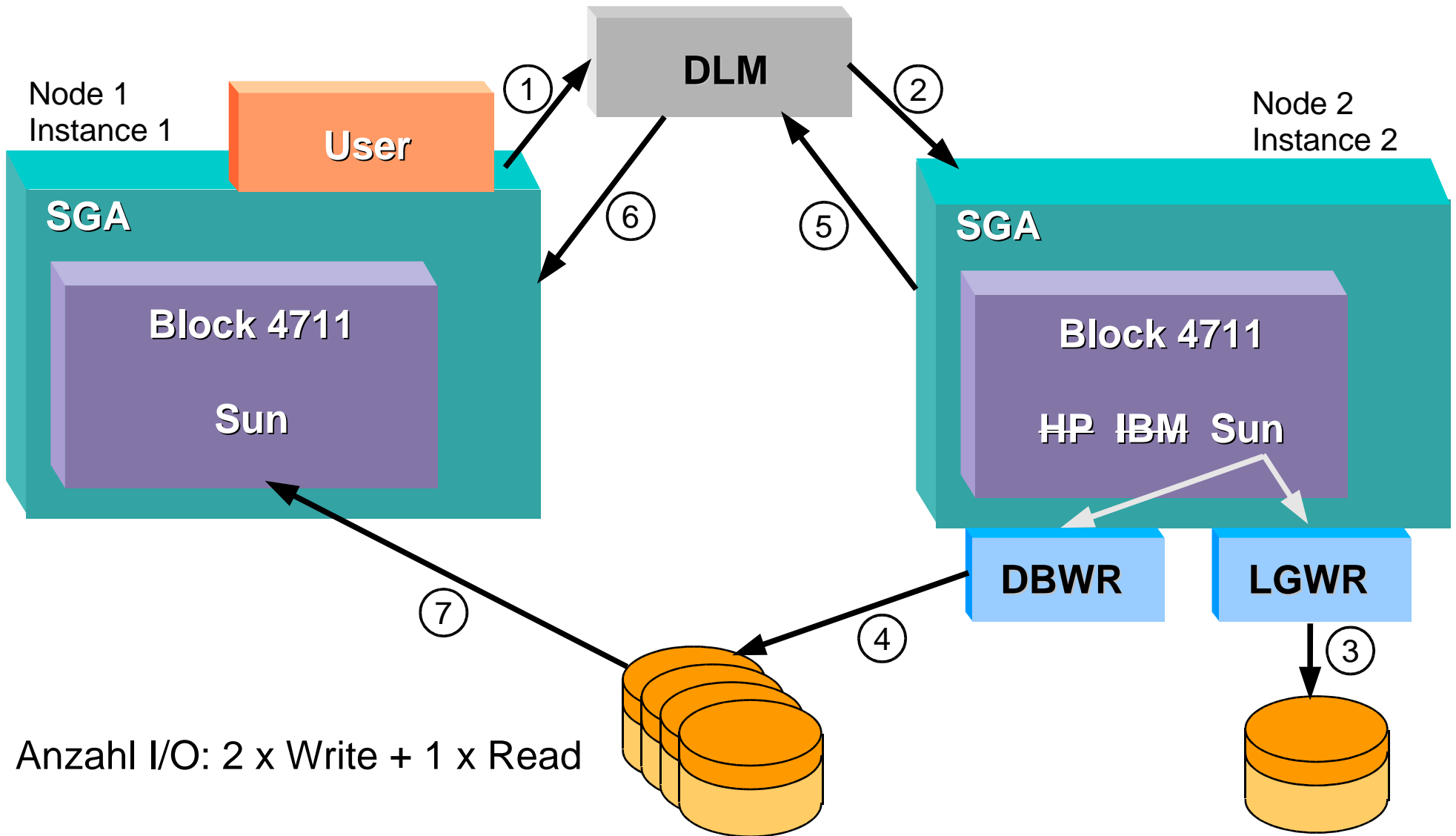
- **Comparisson HA**
 - HA Ziele, DataGuard, HA Oracle, RAC
- **RAC Features**
 - Cache Fusion, TAF, Load Balancing
- **RAC on Solaris**
 - SPARC, x86
- **RAC on Linux**
- **Konfiguration**
 - „best Practice“ - Scaling

RAC Features: Cache Fusion

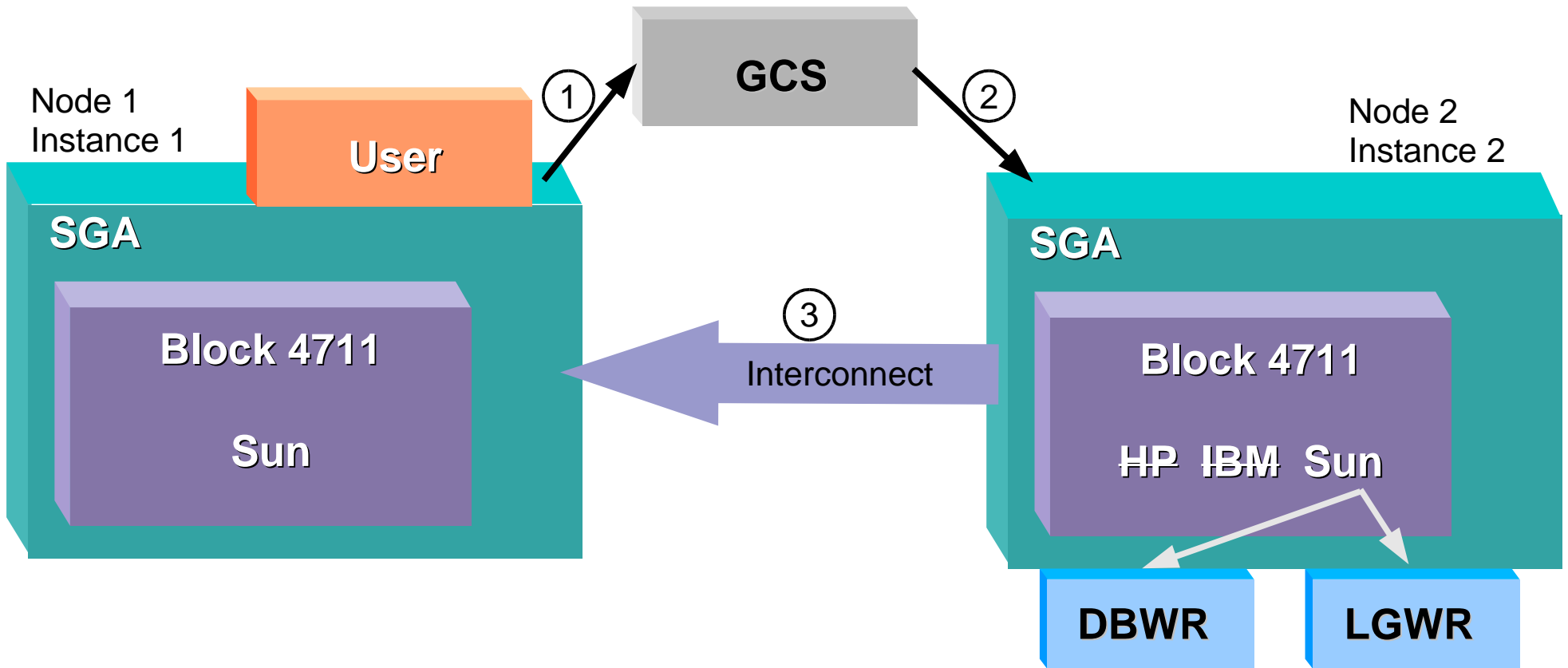
- Optimierung des *Cache Coherency* Protokolls
 - Reduzierung der Anzahl Messages
 - Vermeidung von I/O
- Eindruck einer global SGA (Shared Global Area)



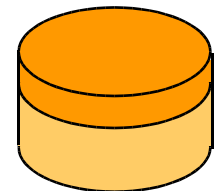
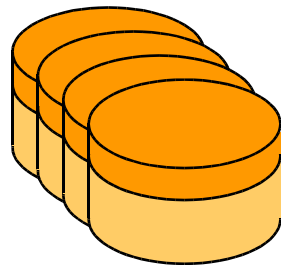
RAC Features: OPS Block Ping



RAC Features: RAC Block Shipping

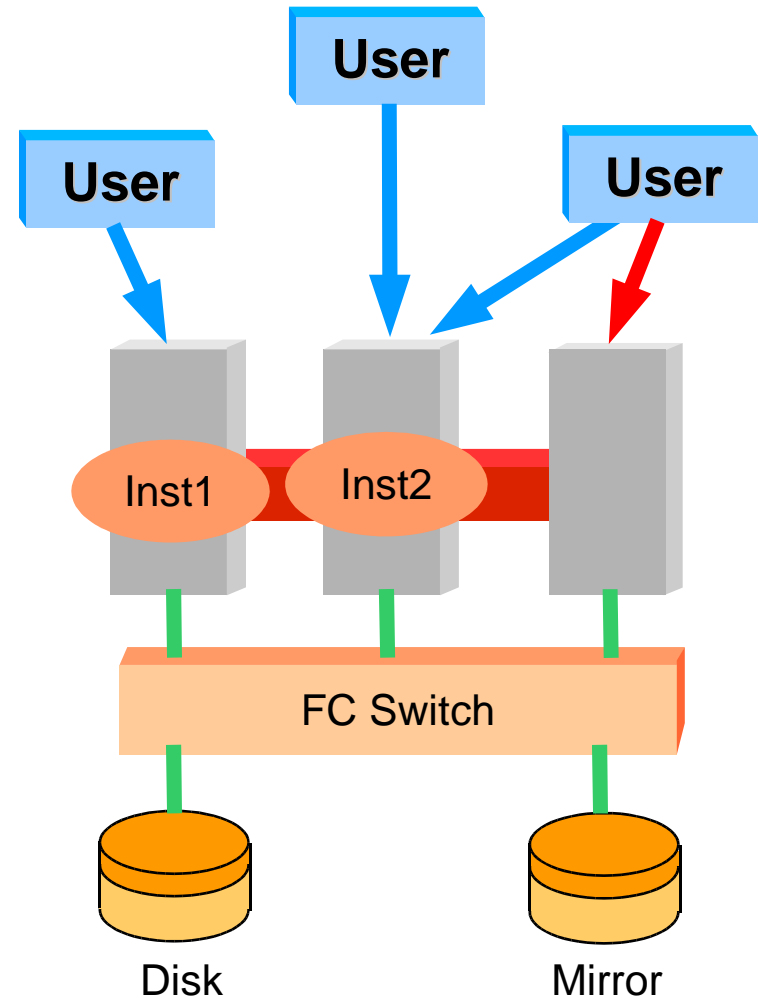


Anzahl I/O: 0 x Write + 0 x Read

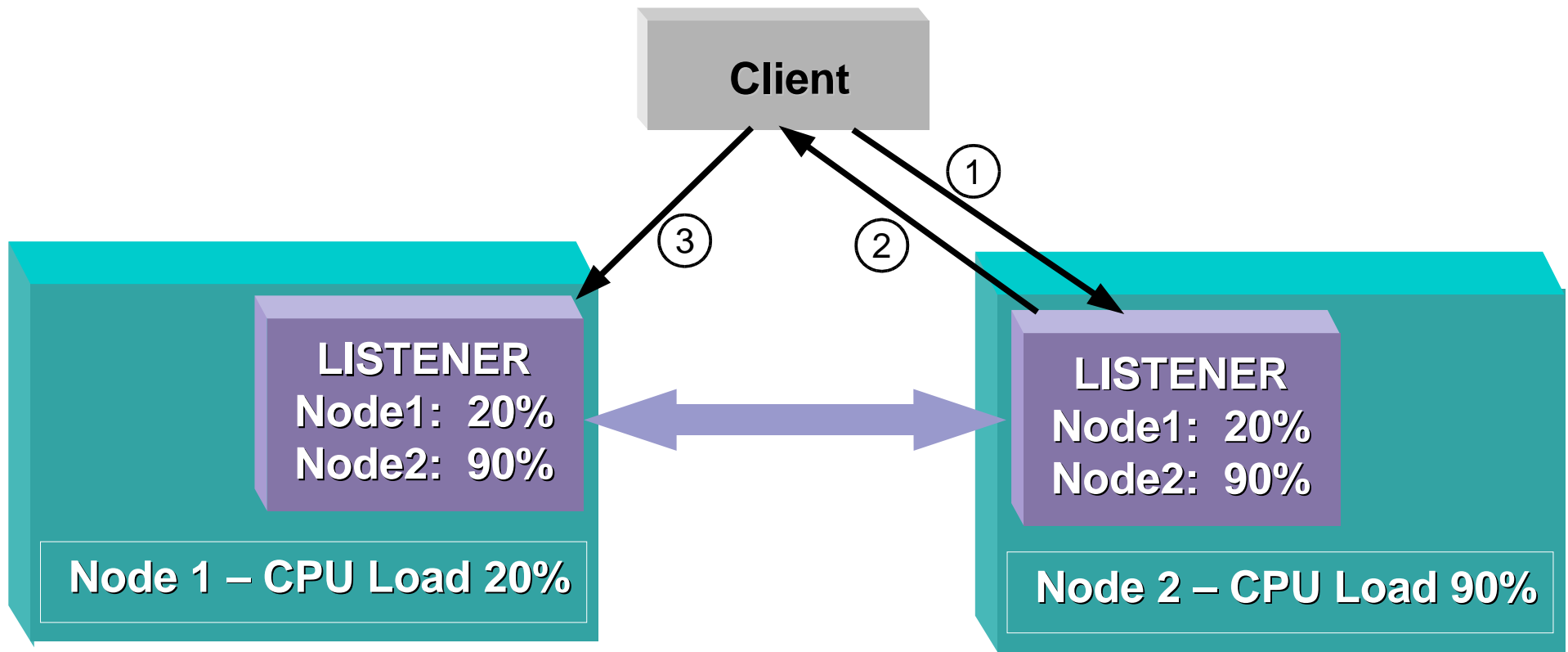


RAC Features: Total Application Failover (TAF)

- Applikationen und Benutzer werden bei Systemausfall automatisch und transparent mit der nächsten laufenden Instanz verbunden
- Es sind „nur“ wenige Benutzer betroffen. Hier sind es 1/3 aller Benutzer
- Übernahme der Benutzer liegt bei weniger als 1 Minute
- Client pre-connect möglich, um große Anzahl an Benutzern schneller zu migrieren



RAC Features: Connection Load Balancing

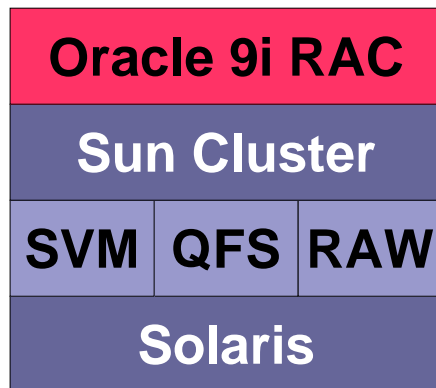


- Listener tauschen sich alle 30 Sekunden über CPU Load aus
- Shared Server:
 - 1. kleinster Node Load
 - 2. kleinster Instance Load
 - 3. kleinster Dispatcher Load
- Dedicated Server:
 - 1. kleinster Node Load
 - 2. kleinster Instance Load

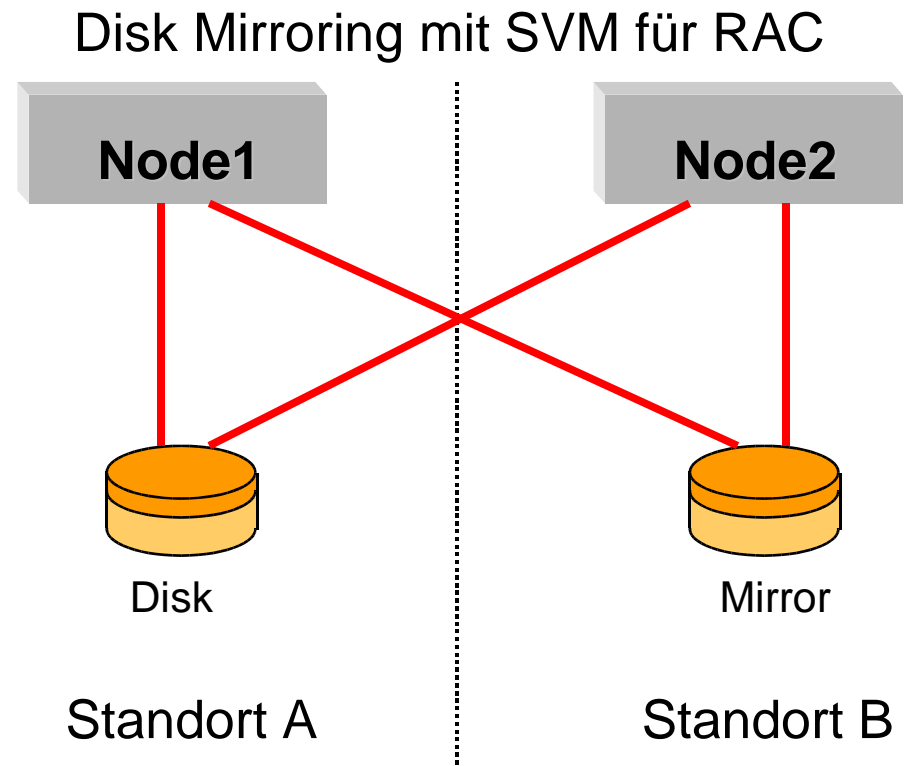
- **Comparisson HA**
 - HA Ziele, DataGuard, HA Oracle, RAC
- **RAC Features**
 - Cache Fusion, TAF, Load Balancing
- **RAC on Solaris**
 - SPARC, x86
- **RAC on Linux**
- **Konfiguration**
 - „best Practice“ - Scaling

RAC on Solaris: Architecture for 9i RAC

- Oracle 9i basiert auf Cluster Framework
- DB kann im Filesystem (QFS) liegen
- Mirroring über Standorte mit SVM
- Bis zu 8 Nodes

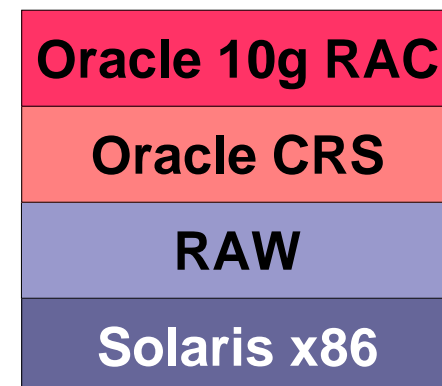
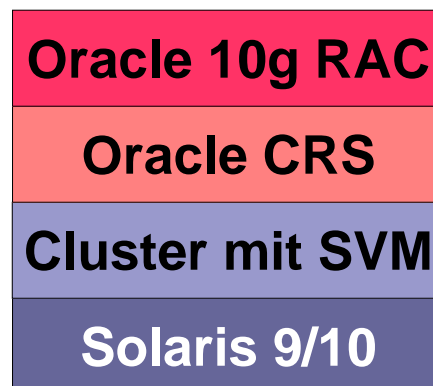
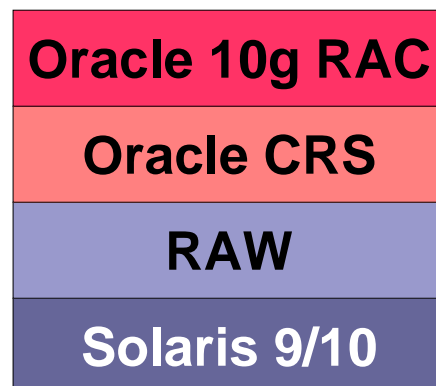


Architecture

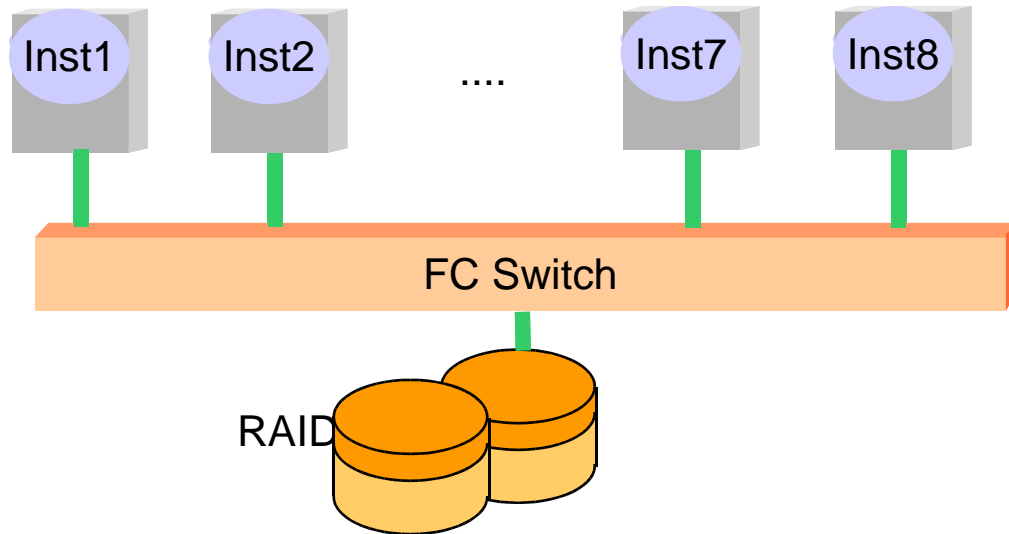


RAC on Solaris: Architecture for 10g RAC

- Oracle 10g bringt sein eigenes „stupides“ Cluster Framework mit
- Cluster Ready Services (CRS) werden per inittab restarted
- DB Instanzen werden durch CRS gestartet und überwacht (kein Cluster Agent)
- RAW Devices können bequem mit SVM Partitioning angelegt werden

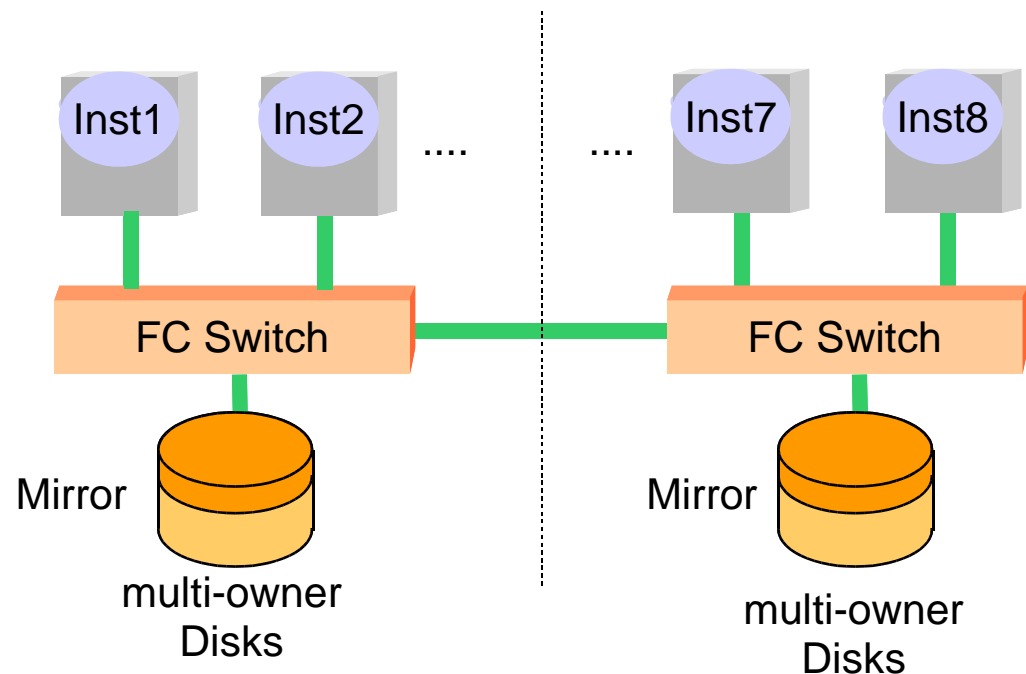


RAC on Solaris: Sun Cluster RAC SVM Edition



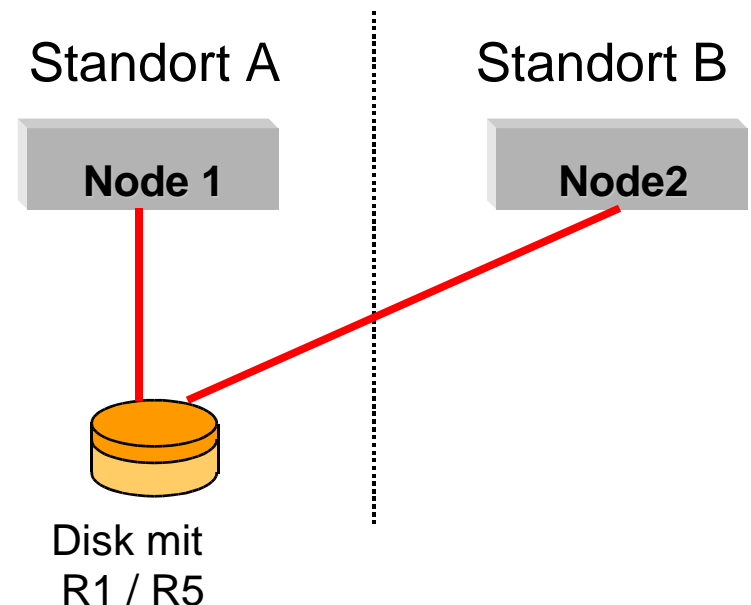
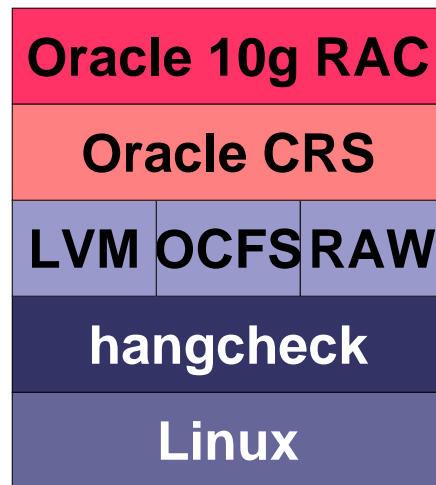
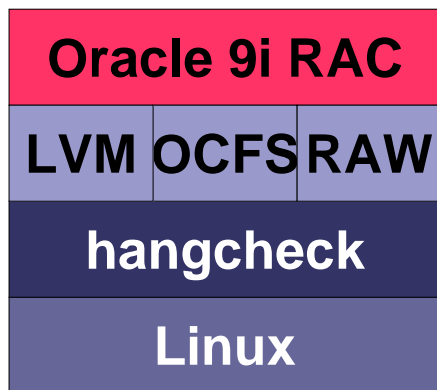
Rechner können an 2 Standorten sein, aber greifen auf dasselbe Storage zu, das nur an einem Standort sein kann

Rechner und Storage können an 2 Standorten verteilt werden. Die Spiegelung ist host-based mit SVM multi-owner Disks.



RAC on Linux: Architecture

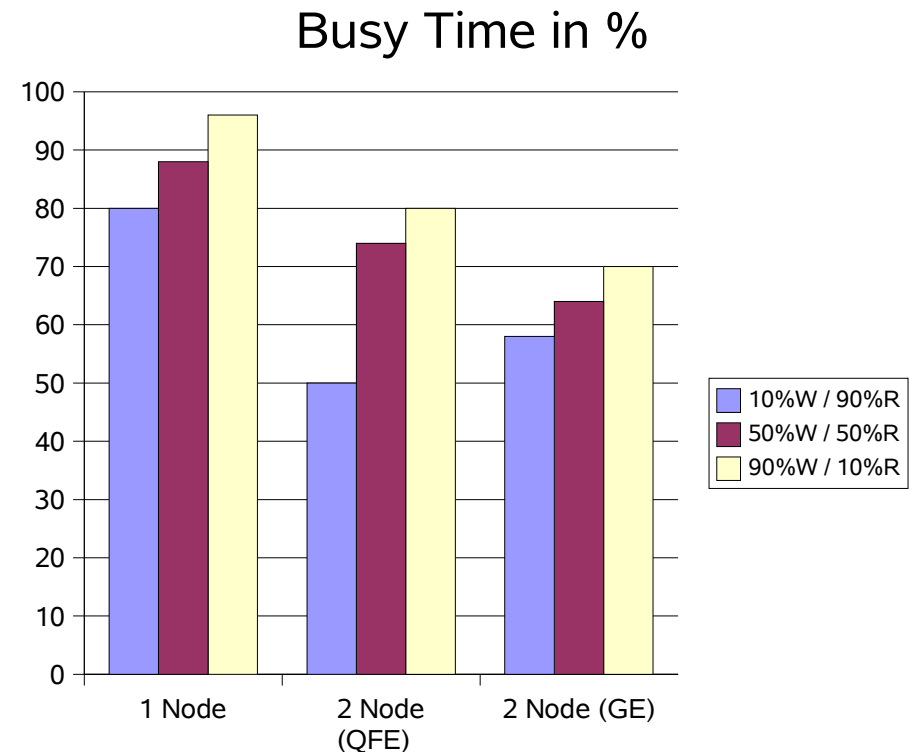
- RAC on Linux läuft mit RH ES/AS oder SLES8+9
- DB Files können nur von der HW gespiegelt werden, d.h keine standortübergreifende Spiegelung mit LVM möglich
- DB Files können im Filesystem mit OCFS liegen – keine Binaries und Logs
- Nur 2 Node RAC mit OCFS; kein OCFS Support für SE RAC (4CPUs)



- **Comparisson HA**
 - HA Ziele, DataGuard, HA Oracle, RAC
- **RAC Features**
 - Cache Fusion, TAF, Load Balancing
- **RAC on Solaris**
 - SPARC, x86
- **RAC on Linux**
- **Konfiguration**
 - „best Practice“ - Scaling

Konfiguration : “best practise“ - Skalierung

- Availability durch horizontale Skalierung
- Performance durch vertikale Skalierung
- Viele kleine Server produzieren Overhead
 - Systemload aller Server steigt
 - Erfahrung: 1 Server 80%, 2 Server 50%
-> 20% Overhead bei Lastverteilung
- Empfehlung:
 - ++ wenige große Server
 - viele kleine Server



Konfiguration : “best practise“ - Transparent Application Failover

- Scenario 1
 - shutdown abort von Instance 1
 - DB Connect geht automatisch und transparent zum nächsten Knoten
 - Failover < 1min
- Scenario 2
 - Power off von Knoten 2
 - DB Connect bzw. Client bekommt nichts vom Ausfall mit
 - TCP/IP Client Timeout Parameter laufen ab, erst jetzt ist der Ausfall bekannt
 - Failover findet nach 11min statt
- Lösung zu Scenario 2
 - Threshold 1: tcp_ip_abort_interval (default: 480000 ms)
 - Threshold 2: tcp_ip_abort_cinterval (default: 180000 ms)

Diskussion:

- Fragen