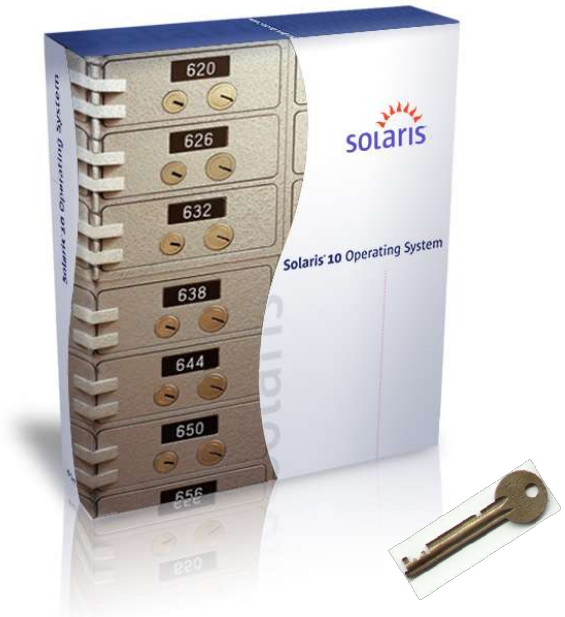
Best Practices für Solaris Container

Franz Haberhauer
OS Ambassador
Plattform Technologie Team
Sun Microsystems GmbH

Solaris Container

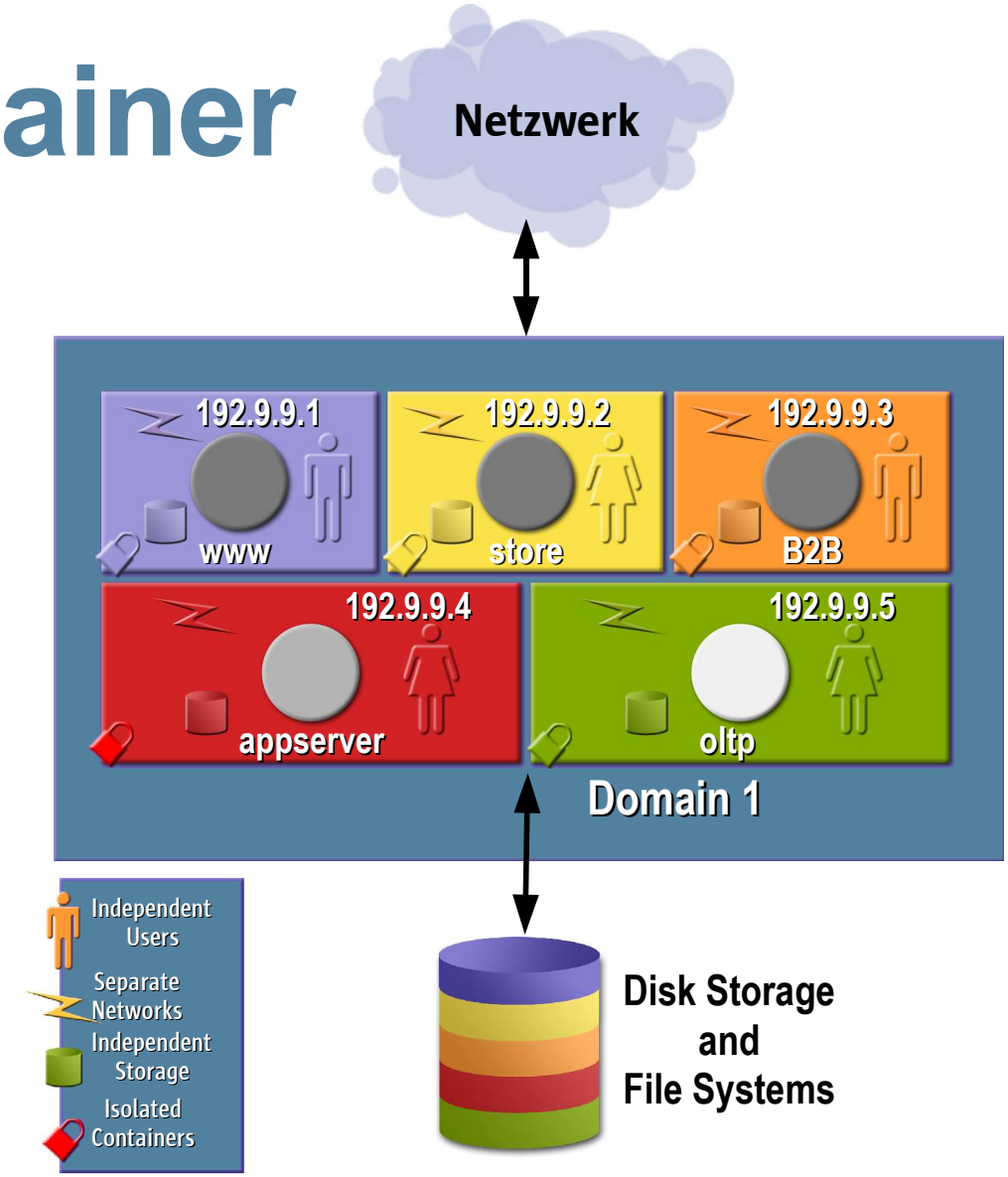


Resource Management

+

Isolation im Hinblick auf

- **Security**
- **Fehler**
- **Software-Lizenzierung**



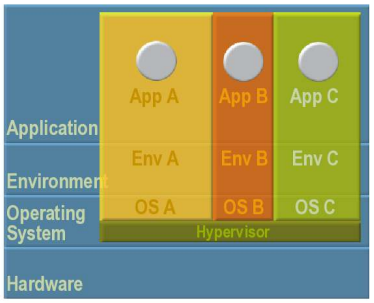
Technologien zur Servervirtualisierung

Hardware-Konsolidierung

Hardware-Partitionierung

Dynamic System Domains

Virtuelle Maschinen



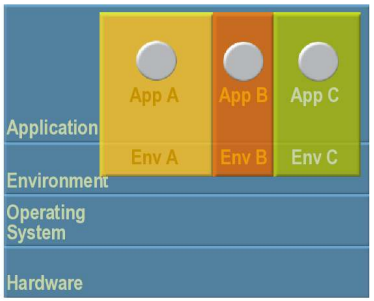
VMware Xen

- Mehrere OS-Instanzen
  - > Administrations-Ansätze aber auch Aufwand unverändert
  - > (Lizenz-)kosten pro OS-Instanz

Mehr Flexibilität

OS-Konsolidierung

Virtuelle Umgebungen



Solaris Container (Zonen + SRM)

Resource-Management

Solaris Resource Manager (SRM)

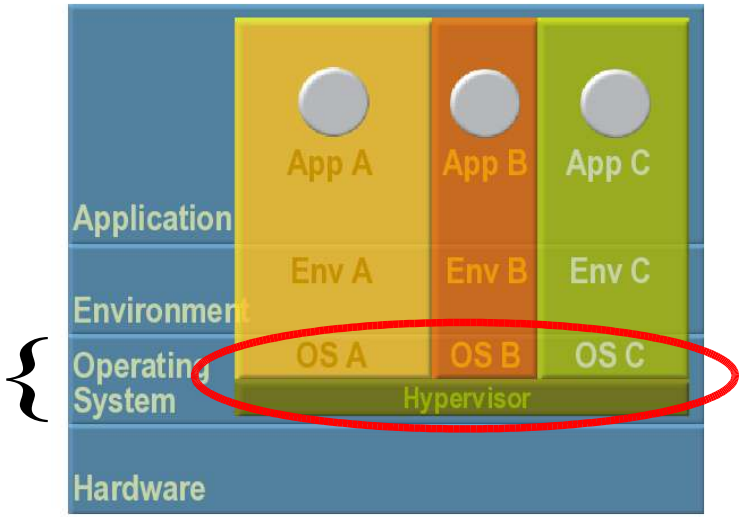
- Nur eine OS-Instanz
  - > Weniger zu verwaltende Instanzen
  - > Zonen Teil der OS-Instanz
- Beste Effizienz, da keine zusätzliche Virtualisierungsschicht

Stärkere Separierung

Virtual Maschinen vs. Container

5-15+% und mehr Overhead

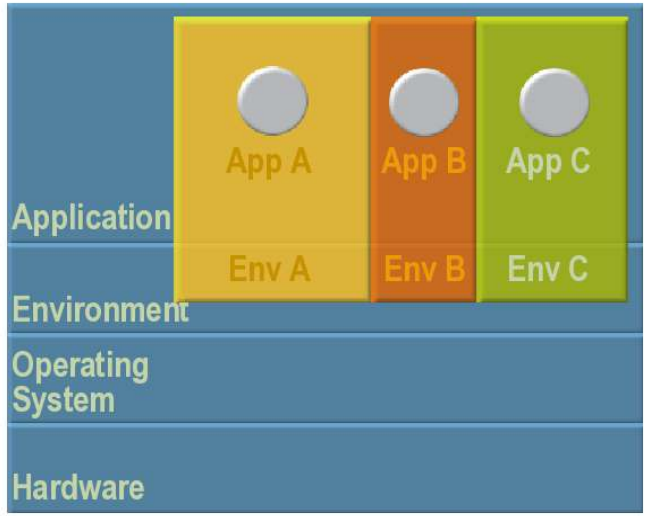
Several OS Instances



IBM LPAR
HP VPAR
VMware
Xen

<1% Overhead
with 4000 tested on a V880

Single OS Instance

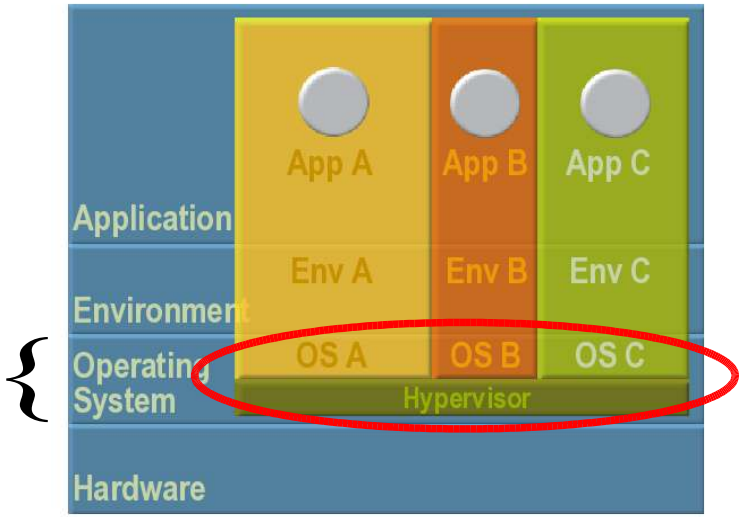


solaris™

Virtual Maschinen vs. Container

5-15+% und mehr Overhead

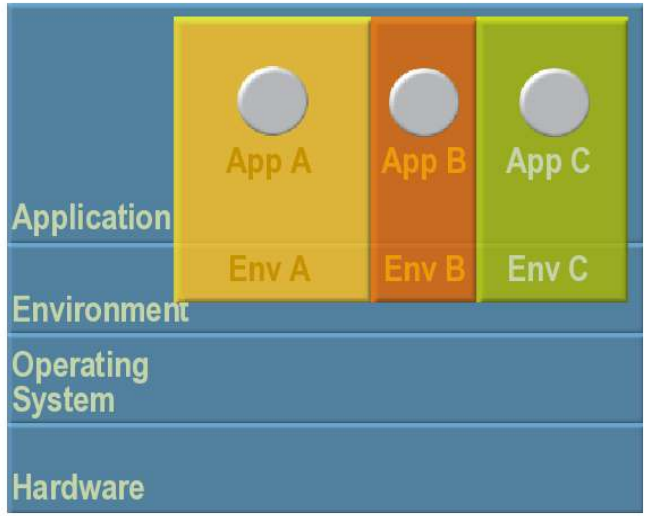
Several OS Instances



IBM LPAR
HP VPAR
VMware
Xen

<1% Overhead
with 4000 tested on a V880

Single OS Instance



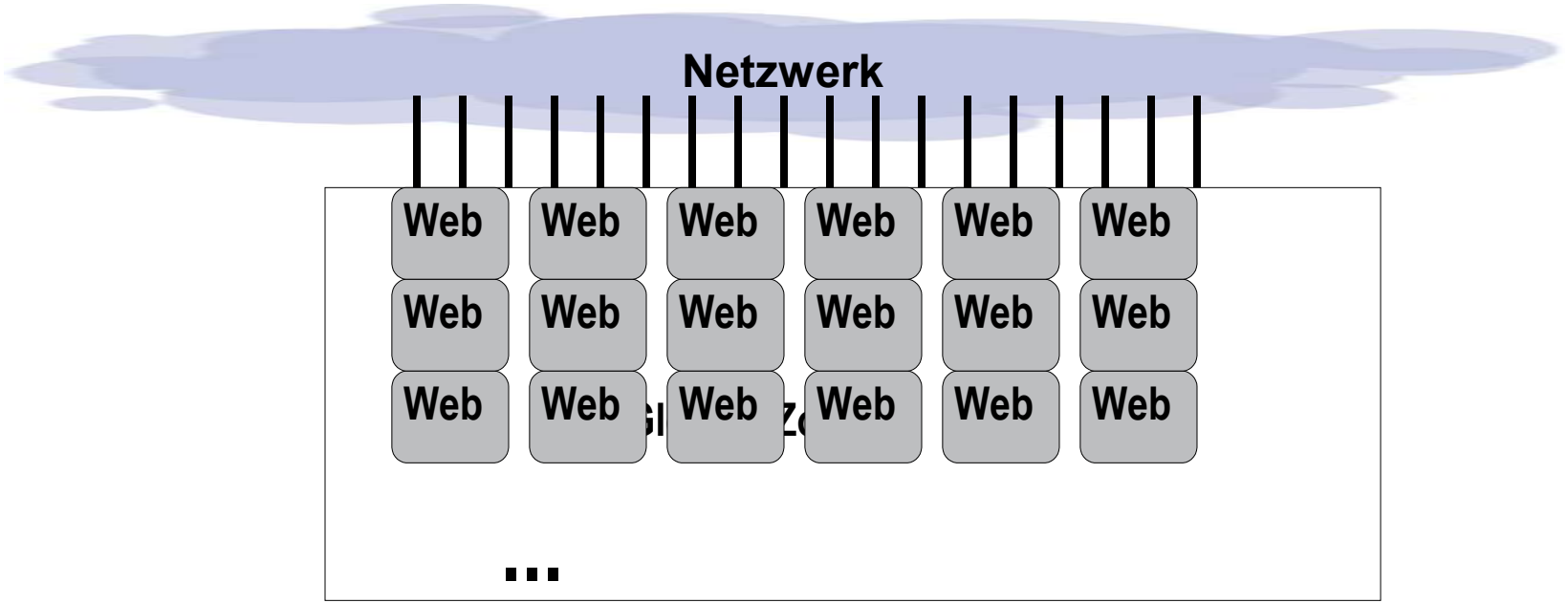
solaris™

Anwendungsfälle für Container

- Hosting unterschiedlicher Umgebungen
  - Web-/App-Server, Email-Server
- Konsolidierung von Anwendungen
  - zur Optimierung der Auslastung
- Separation von Anwendungen
  - Security-Isolation
  - Virtualisierung von Anwendungen
- Software Entwicklung
  - Entwicklung/Test/Qualitätskontrolle/Produktion

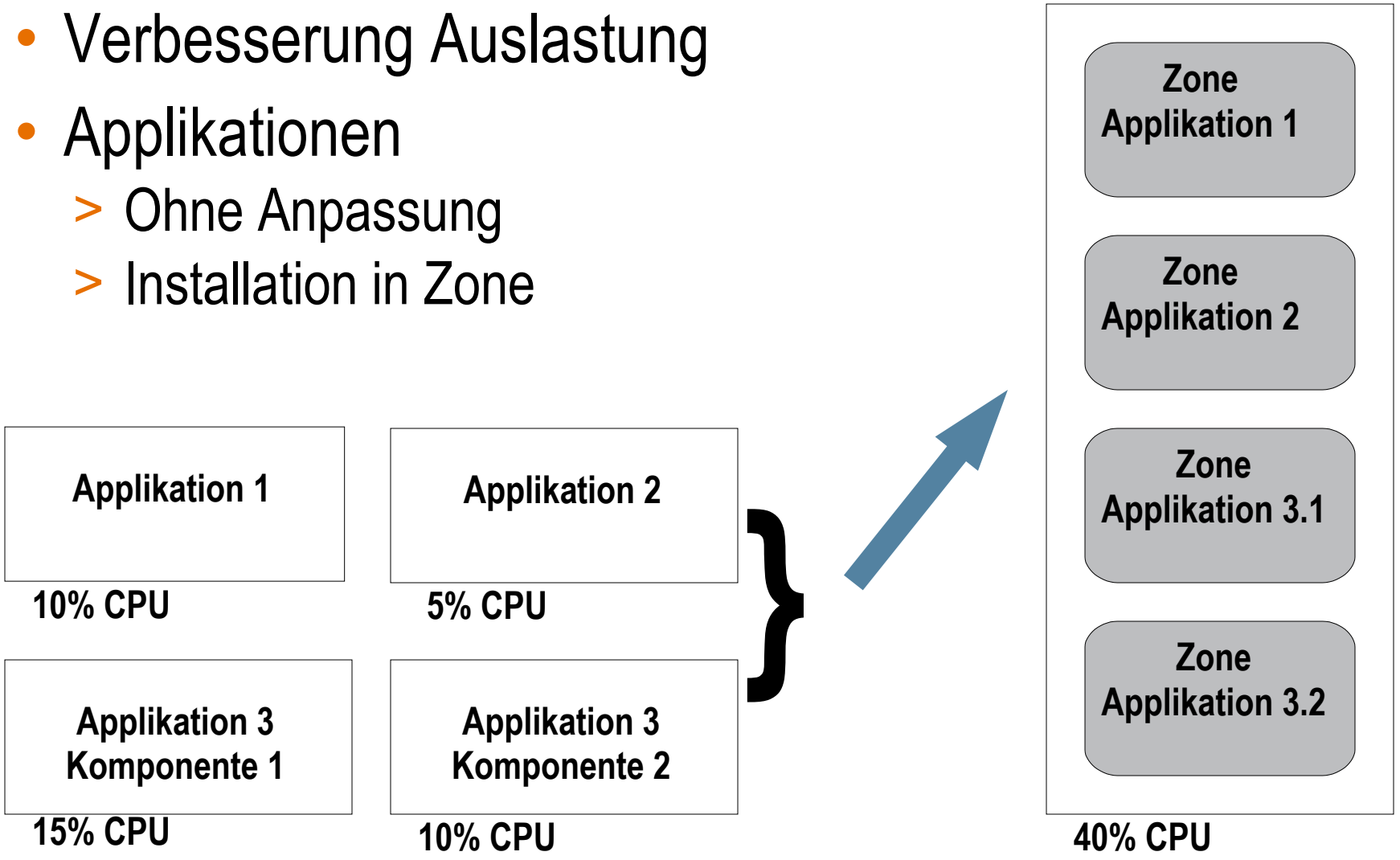
Hosting unterschiedlicher Umgebungen

- Schnelle, automatisierte Provisionierung
- einheitliche Zonen
- Backup/Restore vereinfacht
- Root Access



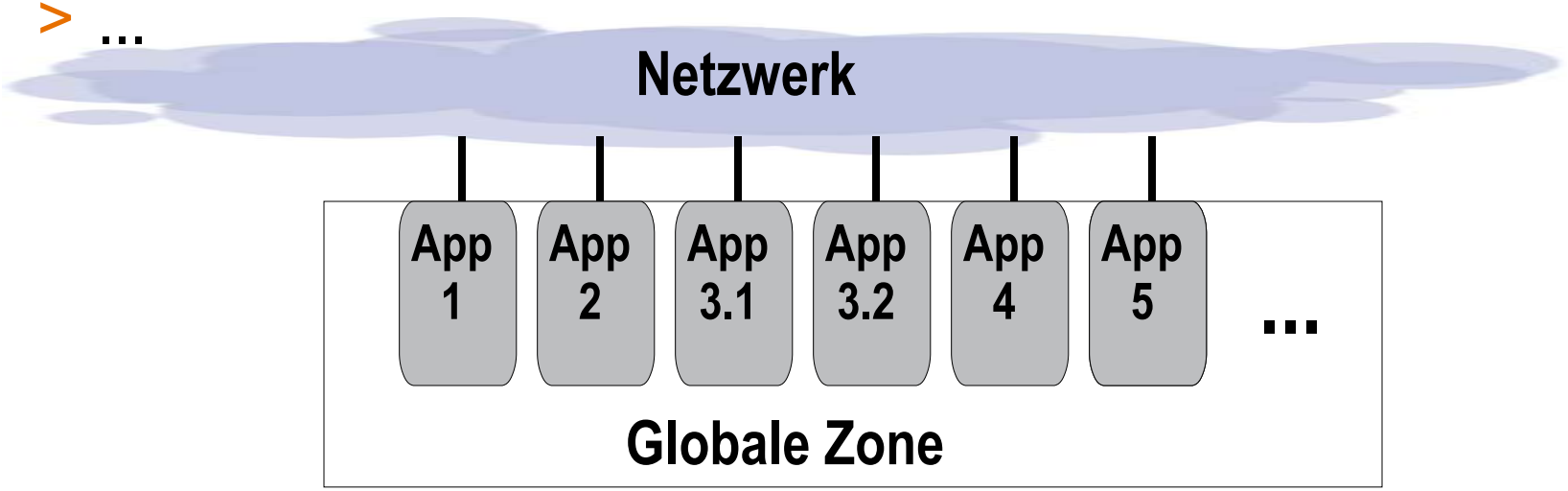
Konsolidierung von Anwendungen

- Verbesserung Auslastung
- Applikationen
  - > Ohne Anpassung
  - > Installation in Zone



Separation von Anwendungen

- Eigene IP-Adresse
- Auflösung von Abhängigkeiten möglich
  - Directories
  - Service Einträge in /etc/services
  - User-Namen
  - ...

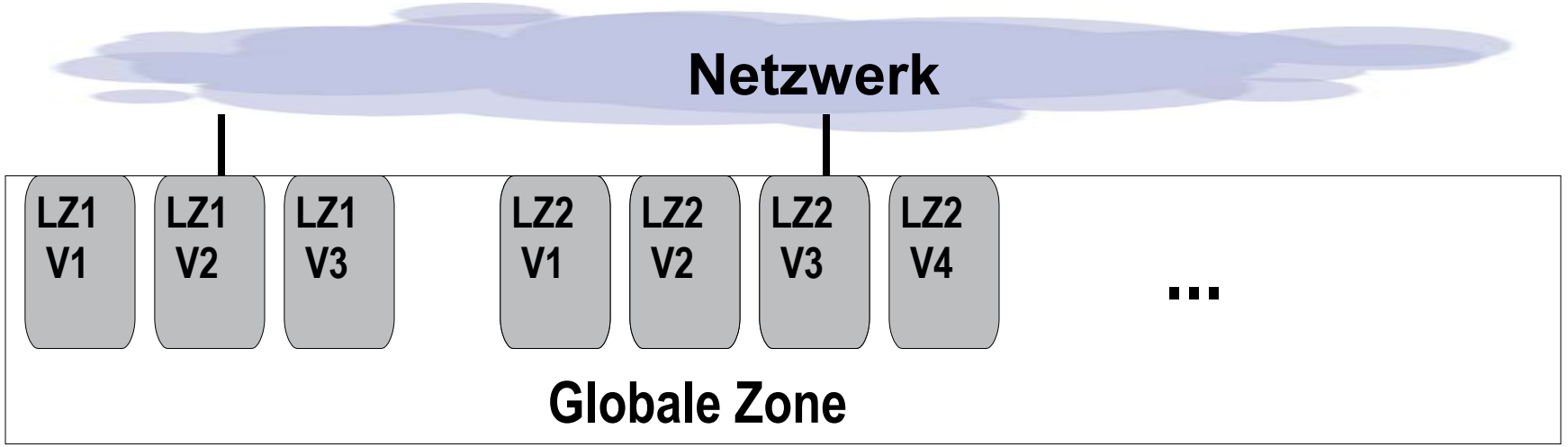


Lizenzoptimierung

- Container als von ISVs anerkanntes Partitionierungskonzept zur Beschränkung von kapazitätsorientierten Lizenzen
  - Oracle “Capped Containers”
    - Container an Prozessorset gebunden

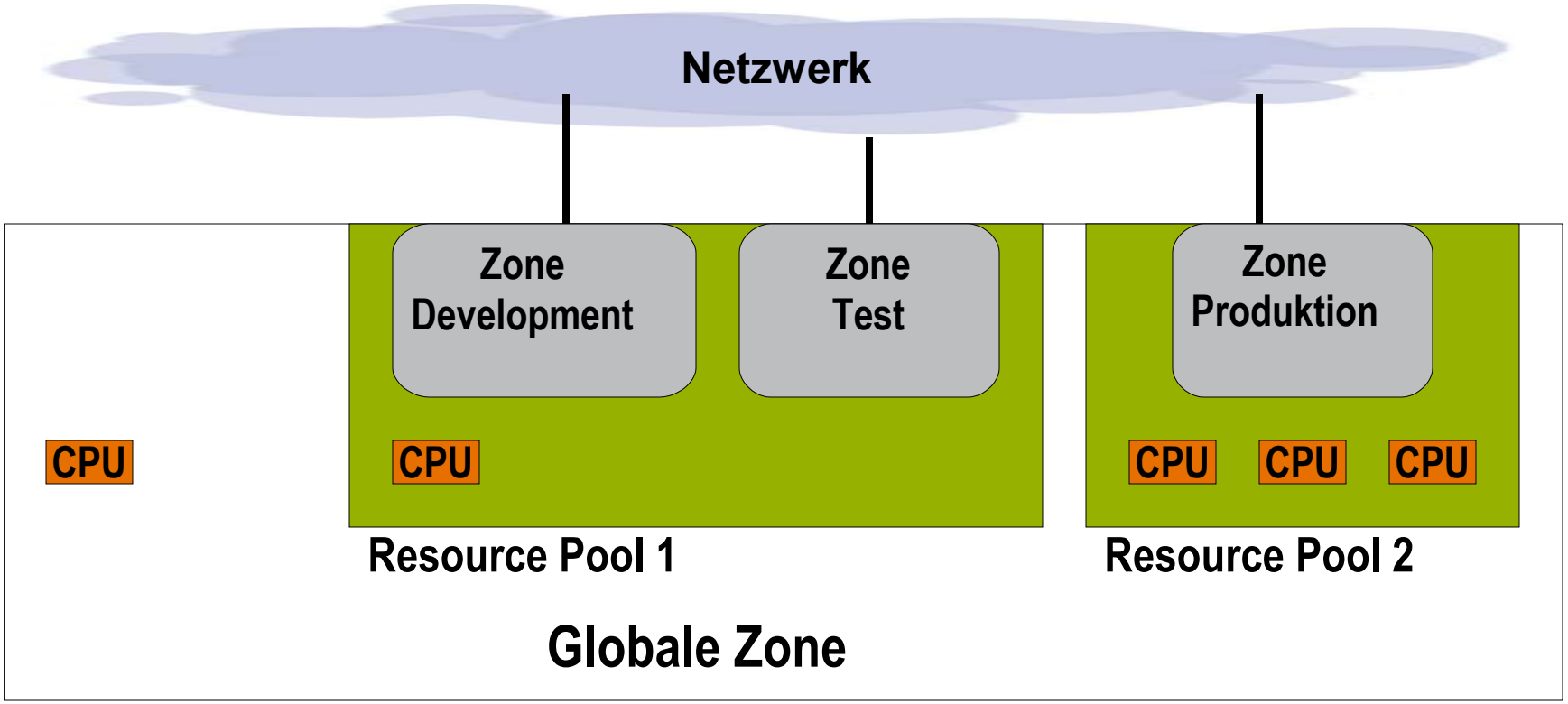
Software Entwicklung

- Verschiedene Software Versionen
  - > Middleware, eigene Software
- Multi-Tier Anwendungen auf einem System (Dtrace!)
- Konsolidierung auf wenige Testrechner
  - > Lastspitzen sind entkoppelt
- Alte Versionen in Zonen im Zugriff (ggf. gleiche IP)



Development, Test, Produktion

- Konsolidierung auf einem/wenigen Rechner(n)
- Migration: Test -> Produktion



Schritte zur Einführung von Zonen

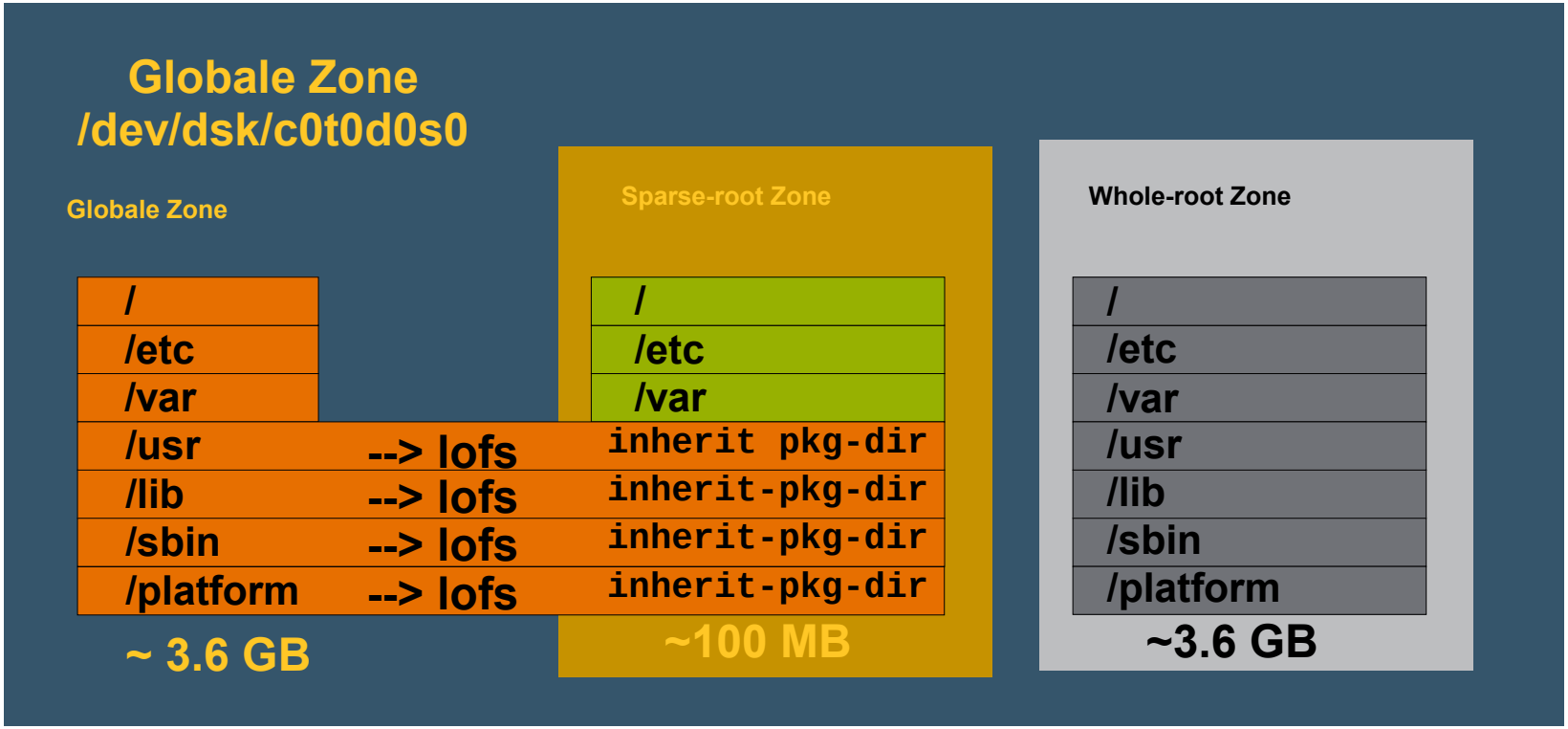
- Ziele für die Einführung klar machen
  - Wo können Zonen helfen ?
  - Welches Ressource Management ist erforderlich ?
- Anwendung auf Solaris 10 verfügbar ?
  - Besonderheiten der Anwendung in der Zone bekannt ?
- Erfahrungen mit Solaris 10 verfügbar ?
  - Neue Features (SMF)
  - Veränderte Betriebskonzepte

- Der 1. Schritt: Einführung von Solaris 10 !
- Der 2. Schritt: Einführung von Zonen

Deployment von Zonen

- Zonentyp festlegen
  - Sparse-root Zone vs. Whole-root Zone
- Root-Plattenlayout
  - Größen, inherit vs. mount, Raw vs. Filesystem
- Verändertes Servicemodell mit Zonen
  - Zonenorientiert oder Serviceorientiert
- Veränderte Betriebskonzepte mit Zonen
  - Zonen in Netzwerken
  - Installation, Abnahme, Monitoring, Backup, Patching, etc.

Zonentypen
Sparse-root und Whole-root Zonen



Zonentypen
Wann, Was ?

- sparse-root Zone (empfohlen)
  - einfache Administration
  - schnelle Installation
  - reduzierter Plattenbedarf
  - Weniger Hauptspeicherbedarf durch gemeinsam genutzte Speicherbereiche
  - Softwareinstallation durch Packages in schreibbare Verzeichnisse oder durch tar-Files
- whole-root Zone
  - besondere Schreib-Anforderungen der Anwendung (JES)
  - besondere Patch-Erfordernisse
  - Testzone für Softwareentwicklung, Releasetest

Filesysteme und Zonen

- Root-Filesystem Layout ist wichtiger Anfangspunkt
  - Globale Zone
    - 5-8 GB für /
    - /var extra
    - für Live Upgrade FS der gleichen Größe nochmals vorhalten
  - Was wird zwischen Zonen inherit-pkg-dir ?
    - jedes in der globalen Zone installierte Package wird in jede Zone vererbt (inherit-pkg-dir) oder kopiert (mount)
    - Ausnahme: Installation mit `pkgadd -G`

Jeder Zone ein eigenes Filesystem ?

- Jeder Zone ein eigenes Filesystem
  - ca. 100MB min., /var separat
  - Komplette Separation der Zonen im Filesystem
  - Größen je nach Sparse-root oder Whole-root Zone
  - Zone kann so durch das Filesystem separat gesichert oder bewegt werden
- Mehrere Zonen teilen sich ein Filesystem
  - Kooperatives Ressourcensharing
  - alle / und /var der Zonen zusammen in ein separates Filesystem legen
    - /zones/var/<zone1>, /zones/var/<zone2>
    - /zones/root/<zone1>, /zones/root/<zone2>,...

Zentraler Mount der Filesysteme

- empfohlen
- mount durch die globale Zone
- per lofs der lokalen Zone bereitgestellt
- Read-write oder read-only

```
global# newfs  /dev/rdsk/c1t0d0s6
global# mount  /dev/dsk/c1t0d0s6 /export/opt/local
global# zonecfg  -z  zone1
       add fs
          set dir=/opt/local
          set special=/export/opt/local
          add options ro
          set type=lofs
          end
       exit
```

Filesystem-Mount beim Zoneboot

- durch die globale Zone

```
global# newfs  /dev/dsk/c1t0d0s6
global# zonecfg  -z  zone1
       add fs
          set dir=/opt/local
          set special=/dev/dsk/c1t0d0s6
          set raw=/dev/rdsk/c1t0d0s6
          set type=ufs
          add options ro,nodevices
          end
       exit
```

Filesystem-Mount in der Zone

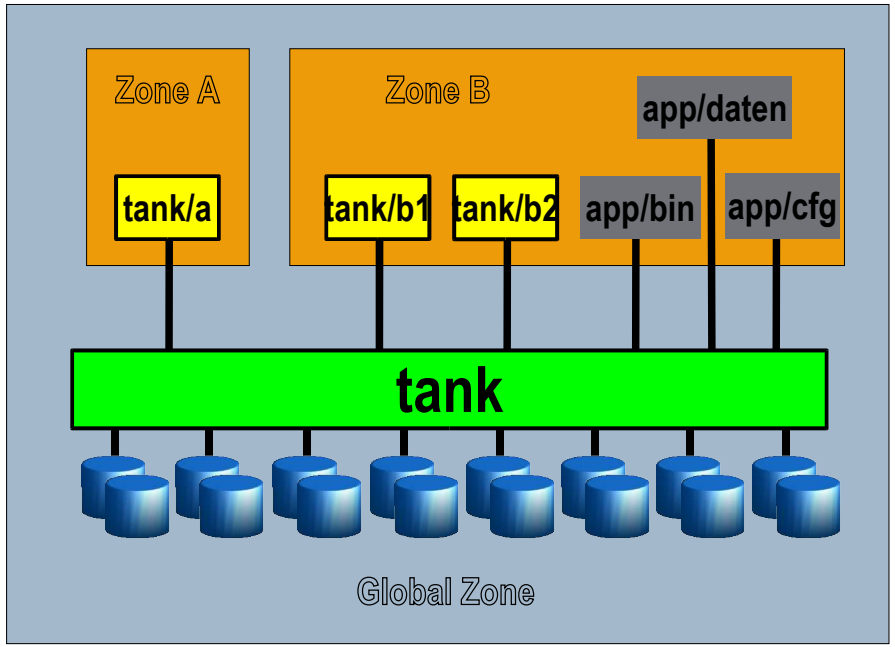
- selten: direkter Zugriff auf Devices

```
global# zonecfg -z zone1
        add device
           set match=/dev/rdsk/c1t0d0s6
        end
        add device
           set match=/dev/dsk/c1t0d0s6
        end

zone1# newfs  /dev/rdsk/c1t0d0s6
zone1# mount  /dev/dsk/c1t0d0s6  /opt/local
```

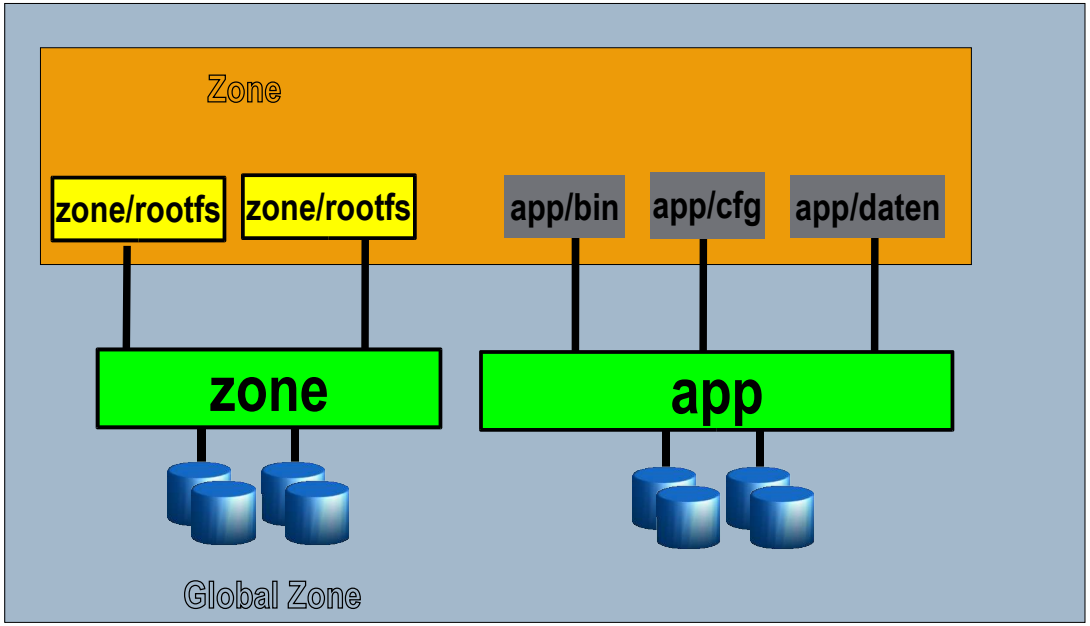
ZFS und Zonen (1)

- Alternativen zur Aufteilung Zonen und ZPools
- Nutzung von Quotas zur Limitierung zwischen /var und /
- Daten und OS teilen sich den ZPool
- Alle Zonen in einem ZPool
  - Ressourcensharing zwischen Zonen
  - Sehr einfach konfigurierbar
  - Quotas fuer ZFS und Zonen
  - Kein separater export für spätere Zonemigration von einzelnen Zonen möglich



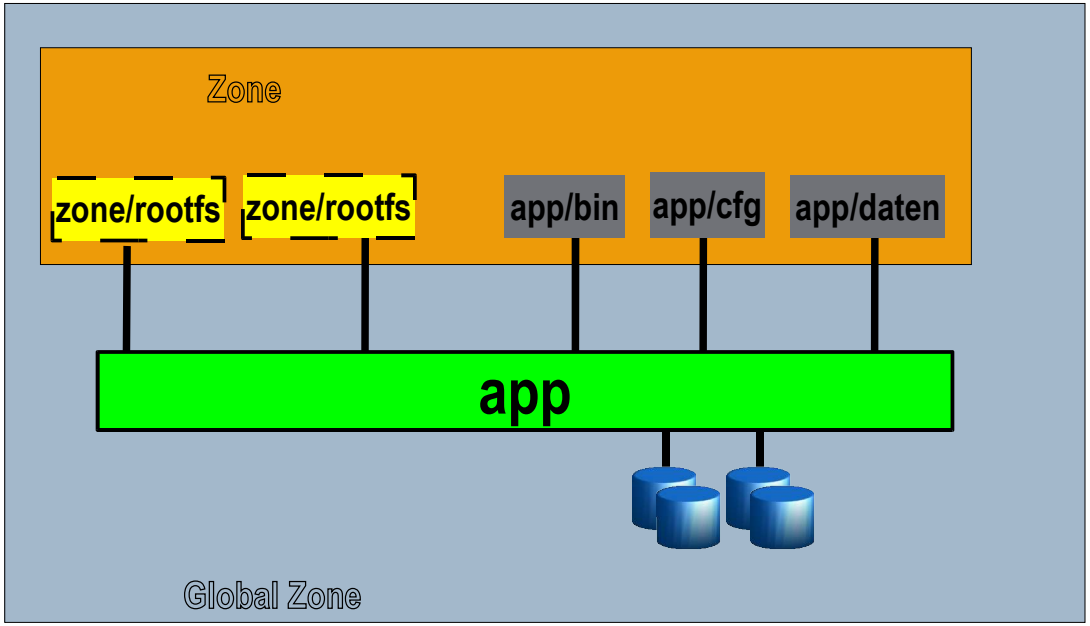
ZFS und Zonen (2)

- **Zoneroot und Anwendungsdaten sind in getrennten ZPools**
  - **jede Zone** zone **hat einen eigenen ZPool**
  - **jede Anwendung** app **hat einen eigenen ZPool**
  - **unabhängige Migration von Zonen zusammen mit der Anwendung**
  - **Anwendung kann separat betrachtet werden**



ZFS und Zonen (3)

- Unabhängige Migration von Anwendungen
- Alle Daten einer Anwendung und der zugehörigen Zone werden zusammen betrachtet
- jede Zone hat einen eigenen ZPool
- jede Applikation hat einen eigenen ZPool
  - root-FS der zone kann bei Bedarf in den Zpool mit hinein

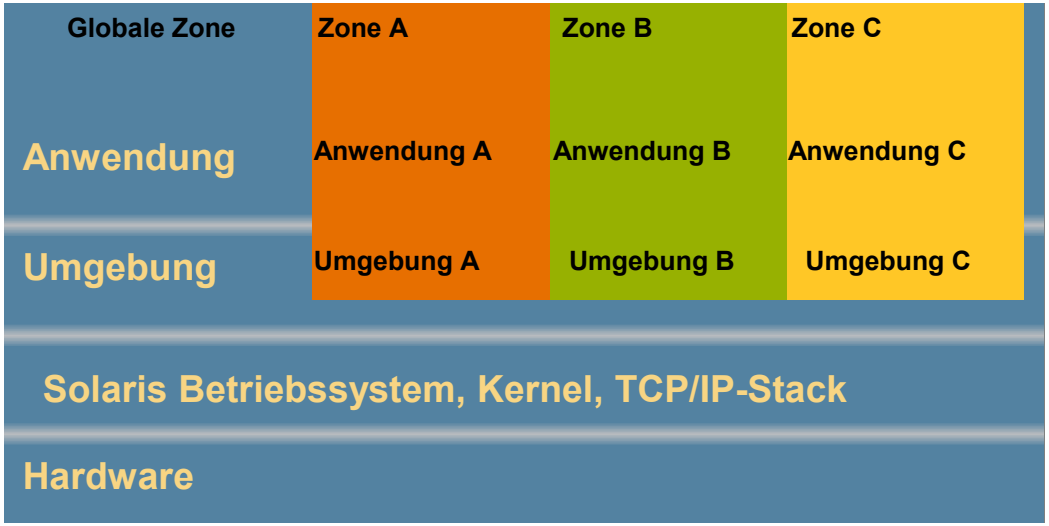


Filesystem: Empfehlung

- Filesysteme nutzen, raw-Devices vermeiden
- mount in globaler Zone + lofs an die lokale Zone
- UFS als Filesystem
  - ZFS, wenn verfügbar
- Wichtige Daten aus der globalen Zone:
  - mit ro-mount der lokalen Zone bereitstellen

Netzwerk und Solaris Zones

- ein TCP/IP-Stack für das gesamte System
- eine Routing Tabelle in dem TCP/IP-Stack
- jede Zone hat eigene TCP/UDP Port Nummern

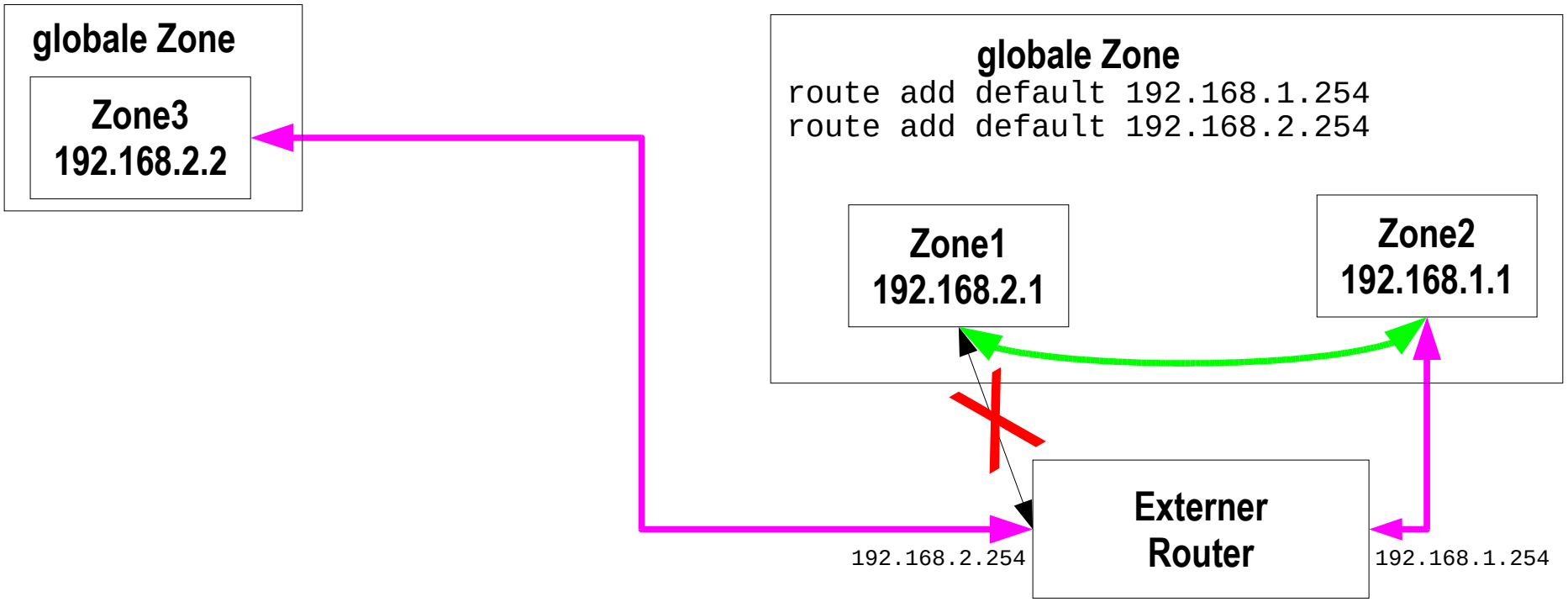


Netzwerk und Zonen

- Nur konfigurierbar in der globalen Zone
  - IPQoS
  - IKE für IPsec
  - IPMP
  - IPFilter
  - Routing
  - NCA
- Nur nutzbar in der globalen Zone
  - DHCP
  - snoop
  - NFS-Server

Routing und Zonen

- Interzone Traffic bleibt immer im TCP/IP-Stack
- Default Gateway in der globalen Zone setzen
  - > mehrere Default Gateways im Solaris



Backup/Restore einer Zone

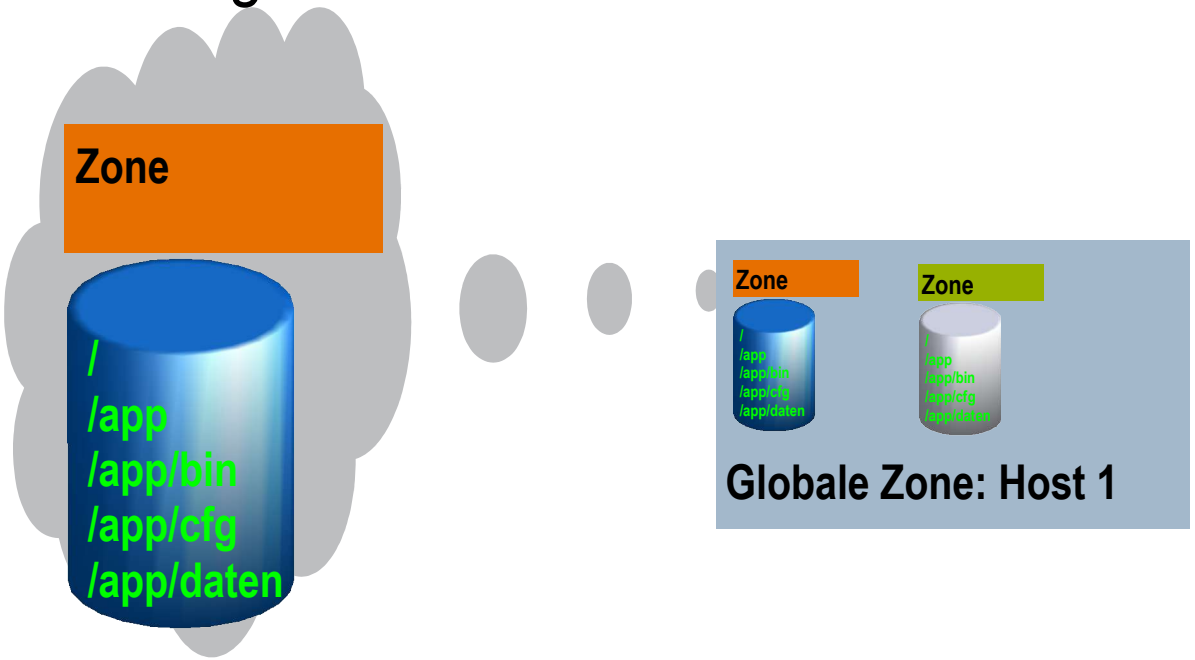
- Backup einer Zone
  - zonecfg export der Konfiguration
  - Backup der zugehörigen Filesysteme incl. zoneconfig
- Restore einer Zone
  - zonecfg import der Konfiguration aus Backup
  - Restore der Filesysteme
- Evtl. auch nur Backup der Zonenkonfiguration und Restore durch automatisierte Reinstallation
  - ggf. zentralisiertes Logging

Nutzung von Backup Tools in Zonen

- inherit-pkg-dir nicht für jede lokale Zone sichern
- Backup aus der globalen Zone für die lokalen Zonen
  - wenn Backup auf Dateiebene sichere /zones/<zone>/*
  - inherit-pkg-dir erkennbar
  - zentrale Backup-config in der globalen Zone
- Backup in der lokalen Zone
  - z.B. Datenbank-spezifische Backupmodule
  - sichere alle sichtbaren Filesysteme
  - inherit-pkg-dir ist nicht erkennbar
  - Backup-config liegt in der Zone selbst
    - u.U. wichtig für spätere Zonenmigration

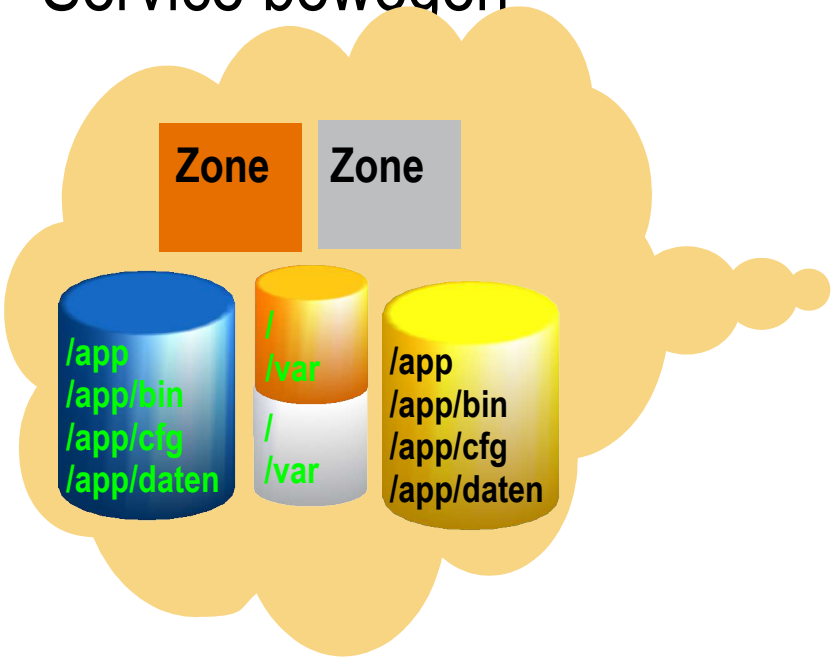
Anwendungen an Zonen gebunden

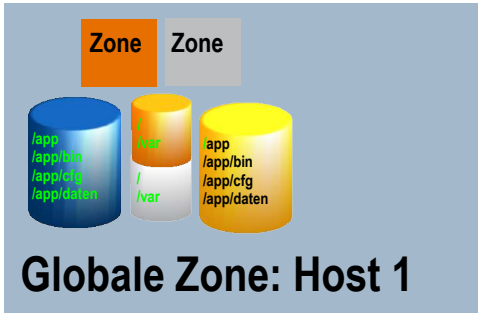
- Zone ist die komplette Anwendungsumgebung
  - > Anwendung ist fest an die Zone gebunden
  - > Trennung der Anwendung bedeutet Neuinstallation
  - > notwendig bei vielen Anpassungen des OS an die Anwendung



Anwendungen unabhängig von Zonen

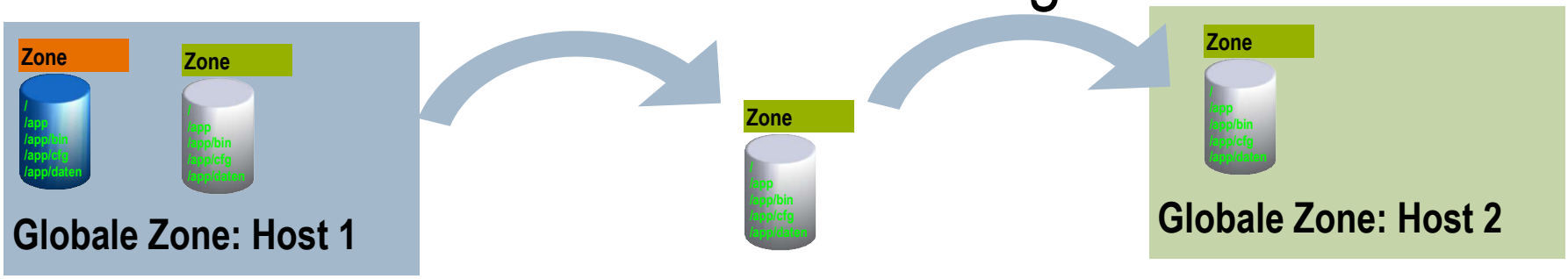
- Zone als neutrale Laufzeitumgebung
  - > Service Orientierte Zone
  - > Service lebt in der Zone, wo das Filesystem ist
  - > Service ist unabhängig von der Zoneninstallation
  - > Service bewegen



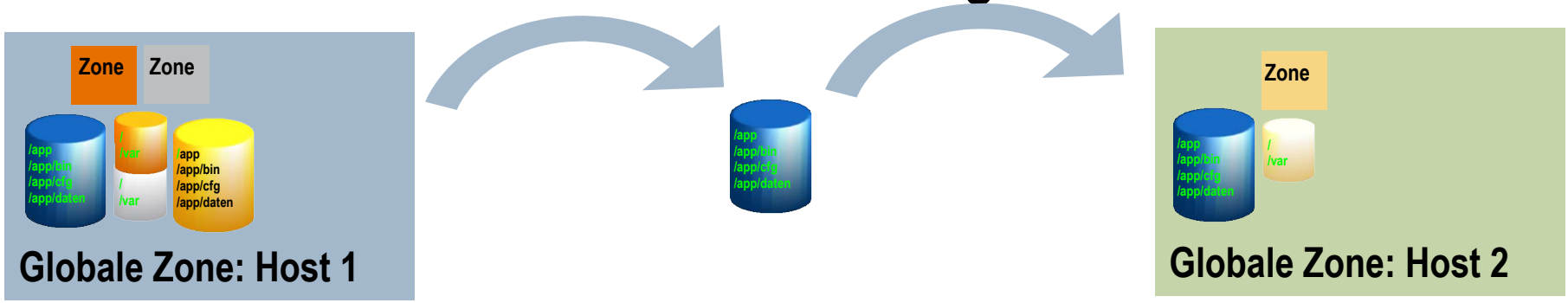


Anwendungen bewegen

- Service wird als Teil der Zone bewegt
- Zonen werden zwischen Hosts mitgeführt



- Service wird mit dem Filesystem bewegt
- Zonen werden bei Bedarf erzeugt



Bewegen eines Service

- SunCluster 3.1 8/05
  - > Migration eines Service als Teil einer Zone
- Künftige Version von SunCluster
  - > Container als “Knoten” zur Ausführung von Services
- Diese beiden Ansätze sind orthogonal

Bewegen einer Zone (Migration)

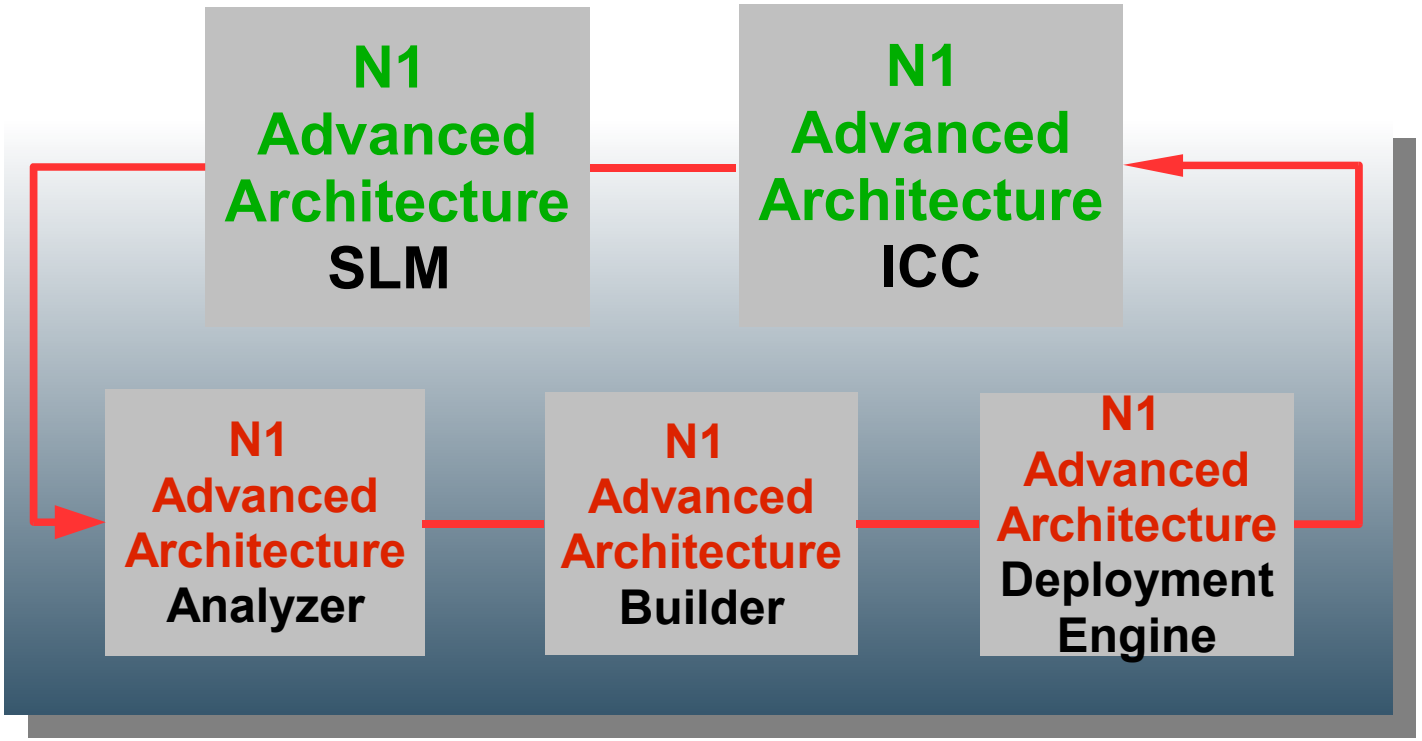
- OpenSolaris Feature (geplant in Sun Solaris)
  - zoneadm detach und attach
    - detach: Status installed --> configured
  - erzeugt <zonepath>/SUNWdetached.xml mit
    - config, Packagelist, Patchlist
- Wichtig:
  - Package- und Patch-Stand in globaler Zone zwischen beiden Systemen müssen gleich sein
  - HW Environment
    - gemountete Devices
    - Interface-Namen
- keine Live Migration

N1 Advanced Architecture für SAP

(jetzt Erweiterung des N1 Service Provisioning System)

SAP Service Level Monitor

Leistungsbasierte Kostenverrechnung

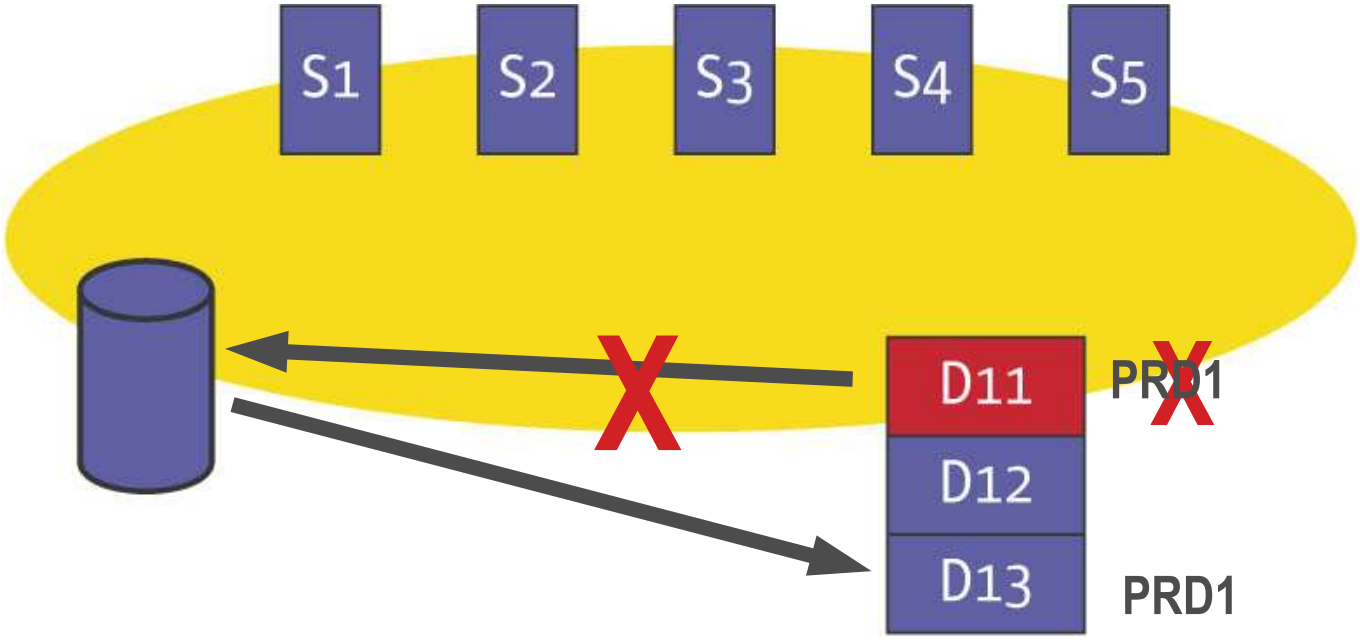


“Infrastruktur DWH”

Werkzeuge zur Standardisierung

Computing power where needed

Dynamisches Verschieben von SAP Instanzen



Deployment Engine steuert Verschieben des SAP Systems von Domain D11 zur Domain D13 und minimiert Schaltzeit und Risiko

Auf Domain D11 laufen einige SAP Instanzen, darunter PRD1.

Auslastung von D11 ist größer 80%

Zur Entlastung von D11 wird SAP Instanz PRD1 auf die freie Domain D13 verlagert

Solaris 10 Ressource Management

- Verwaltbare Ressourcen
  - Netzwerk
    - IPQoS
  - CPU
    - Pools mit Prozessorsets
    - Fair Share Scheduler
  - Hauptspeicher (physikalisch/virtuell)
    - rcapd:(Limitierung des benutzten phys.Hauptspeicher)
  - IPC Ressource Controls
    - prctl vs. /etc/systems-Einträge
  - In Planung
    - Memory Sets*: Nutzbarer Hauptspeicher pro Zone
    - Swap Sets*: Nutzbarer Swap Bereich pro Zone

Anwendungsfälle Prozessor-Sets

- Server Konsolidierung
  - Applikationen bestimmte Prozessoren zuweisen
  - z.B. wenn CPU-Grenzen eingehalten werden müssen
    - z.B. SLA einhalten
    - z.B. Ausgrenzung von Interrupts
- Lizensierung von Software auf CPU-Anzahl
  - nicht von allen Herstellern unterstützt

Container Ressourcen limitieren

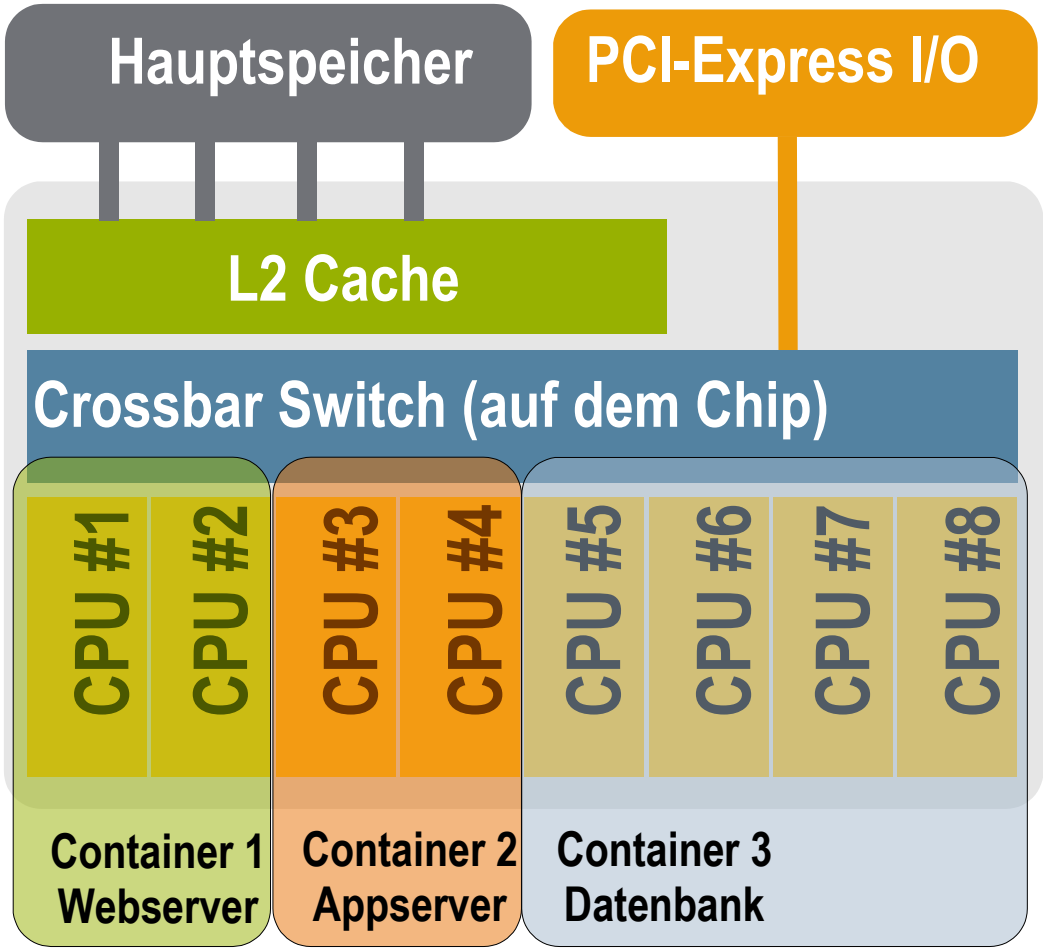
- Begrenzung der benutzbaren CPU-Shares eines Containers
  - zonecfg rctl: zone.cpu-shares
- Begrenzung der max. LWP eines Containers
  - zonecfg rctl: zone.max-lwps (fork-Schleifen vermeiden)
- Dynamische Ressource Pools
  - Ressource Definitionen an Container binden

Ressource Controls mit Zonen

```
# zonecfg -z keetonga
# zonecfg:keetonga> add rctl
# zonecfg:keetonga:rctl> set name=zone.cpu-shares
# zonecfg:keetonga:rctl> add value
  (priv=privileged,limit=20,action=none)
# zonecfg:keetonga:rctl> end
# zonecfg:keetonga> add rctl
# zonecfg:keetonga> set name=zone.max-lwps
# zonecfg:keetonga:rctl> add value
  (priv=privileged,limit=100,action=none)
# zonecfg:keetonga:rctl> end
```

Container und UltraSPARC T1

- Maximale Auslastung und Flexibilität
- Server und Netzwerk alles auf einem Chip
- Container mit Prozessorsets auf Thread-Ebene



Ressource Pools mit Zonen (T2000)

```
# pooladm -e
# pooladm -x
# pooladm -s
# poolcfg -dc 'create pset pool1-pset'
# poolcfg -dc 'create pool pool1-pool'
# poolcfg -dc 'modify pset pool1-pset (uint pset.min = 4; uint pset.max = 4)'
# poolcfg -dc 'transfer to pset pool1-pset (cpu 0; cpu 1; cpu 2; cpu 3)'
# poolcfg -dc 'associate pool pool1-pool (pset pool1-pset)'
# zonecfg -z keetonga
# zonecfg:keetonga> set pool=pool1-pool
# zonecfg:keetonga> exit
```

Literatur

- Solaris Learning Center Learning Center
  - http://www.sun.com/solaris/teachme
- Zones FAQ
  - http://www.opensolaris.org/os/community/zones/faq
- Qualification Best Practices for Application Support in Non-Global Zones
  - http://developers.sun.com/solaris/articles/zone_app_qualif.html
- "Bringing Your Application Into the Zone".
  - http://developers.sun.com/solaris/articles/application_in_zone.html

Literatur
Spezifische Applikationen

- Solaris Container bei der Apache Software Foundation
  - http://www.tbray.org/ongoing/When/200x/2006/03/06/Apache-Server
- Oracle im Container
  - http://www.sun.com/blueprints/0505/819-2679.pdf pp. 22-33
- Sun Cluster Data Service for Solaris Containers Guide
  - http://docs.sun.com/app/docs/doc/819-2664
- Webserver/ftpd und Backup/Restore in einer Zone
  - http://www.sun.com/blueprints/0506/819-6186.pdf pg. 8 und pp. 10-13
- Websphere Application Server im Container
  - http://www.sun.com/software/whitepapers/solaris10/websphere6_sol10.pdf
  - http://blogs.sun.com/roller/page/sunabl?entry=websphere_deployment_on_solaris_10
- Websphere MQ in der globalen und whole-root Zones
  - http://www-1.ibm.com/support/docview.wss?rs=171&uid=swg21233258

Vielen Dank

Franz.Haberhauer@Sun.com
http://blogs.sun.com/FranzHaberhauer
http://blogs.sun.com/Solarium