Solaris™ Containers for System {Admins, Architects, Engineers} and Technical Trainers

Jeff Victor

Senior Technologist, OS Ambassador

Datacenter Practice, USCS

Sun Microsystems, Inc.

Agenda

- Unix Design Centers
- Extensions in Solaris Containers
- Managing Containers
- Uses of Containers

Unix Design Centers

Processes

- Each process has a parent, some have 1+ children
- Each process is owned by a user
  - Ownership determines the ability of the process to do things, enforced by the kernel
- All processes can see each other

Users

- Kernel code allows one user to see and do anything (root)
- All other users are equivalent
- Accounting system included for billing

Files

- One file type
- Everything is a file, has ownership, permissions
  - Files, Directories
  - Devices, Memory
- Kernel enforces security by comparing file owner and perm's against owner of process attempting action
- N-level tree structure with one root “/”

Containers

Design Goals

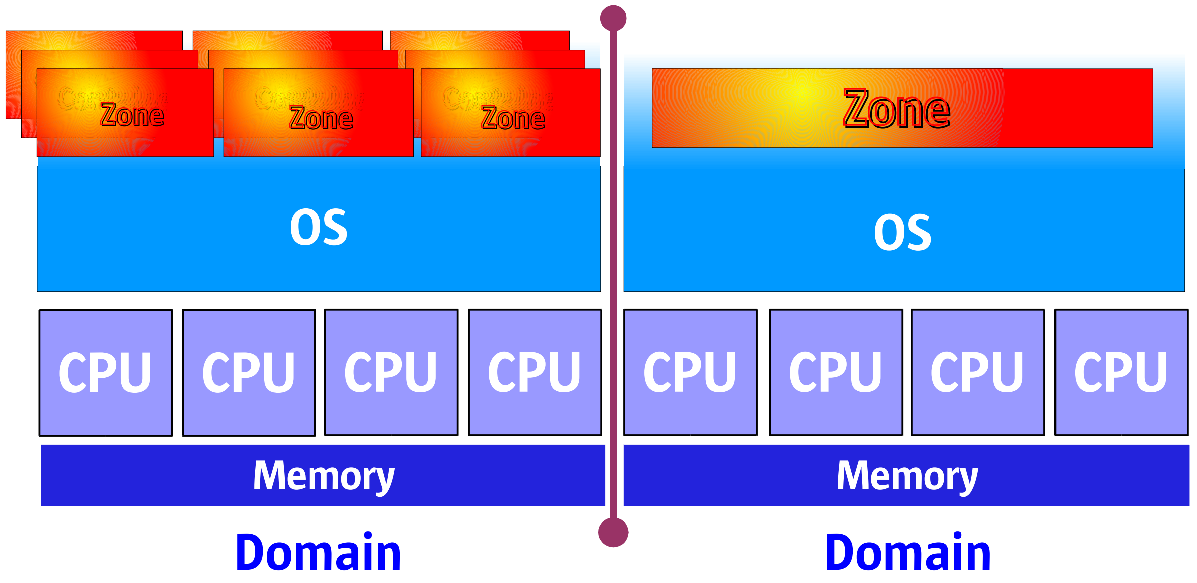
- Multiple isolated pools
  - Security
  - Resource Management
- Comprehensive consolidation facility integrated as a core component of a mainstream OS
  - Should be portable to multiple platforms
- Low administrative overhead
  - Manageable, observeable
  - Reduce, don't increase admin workload
  - Enable delegation of container management
- Low computational overhead
- No additional hardware, licensing or support fees

Containers
Design Decisions

- Support multiple, isolated application environments in one OS instance
  - Achieves isolation, observability, reduced costs
- Software-based solution
  - Achieves portability, observeability, simplicity
- Do not require app changes or recompilation

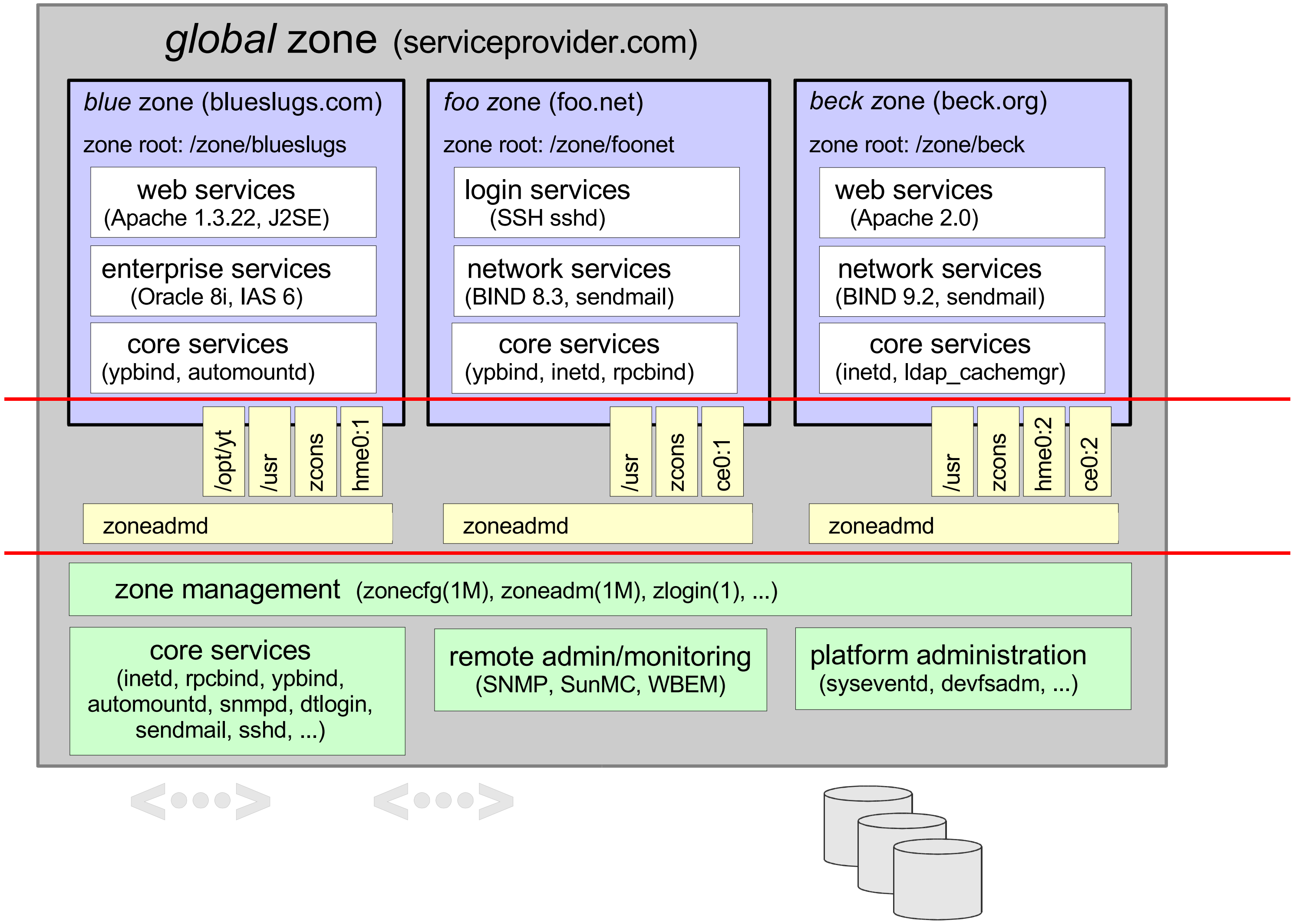
Server Virtualization

Single Solaris instance
- Sub-CPU granularity
- Appearance of many OS instances
- Minimal performance impact



Container Details

Single Application Containers



global zone (serviceprovider.com)
blue zone (blueslugs.com)
zone root: /zone/blueslugs
web services (Apache 1.3.22, J2SE)
enterprise services (Oracle 8i, IAS 6)
core services (ypbind, automountd)
foo zone (foo.net)
zone root: /zone/foonet
login services (SSH sshd)
network services (BIND 8.3, sendmail)
core services (ypbind, inetd, rpcbind)
beck zone (beck.org)
zone root: /zone/beck
web services (Apache 2.0)
network services (BIND 9.2, sendmail)
core services (inetd, ldap_cachemgr)
/opt/yt
/usr
zcons
hme0:1
/usr
zcons
ce0:1
/usr
zcons
hme0:2
ce0:2
zoneadmd
zoneadmd
zoneadmd
zone management (zonecfg(1M), zoneadm(1M), zlogin(1), ...)
core services (inetd, rpcbind, ypbind, automountd, snmpd, dtlogin, sendmail, sshd, ...)
remote admin/monitoring (SNMP, SunMC, WBEM)
platform administration (syseventd, devfsadm, ...)

DC Modifications

Processes

- Each process associated with one container
- Kernel enforces all new rules about inter-container access and control
- Global zone behaves like a non-zoned system
  - Root-owned GZ processes have same powers, across all zones
  - Non-root-owned processes can view info about LZ processes, but cannot signal them
- Processes in one Local Zone (LZ) not visible to any other LZ
  - Root user in an LZ only omnipotent and omniscient within its own LZ, has no power or visibility into other zones
- GZ sees all processes in all zones in one process tree; LZ sees only its own sub-tree

DC Modifications

File System

- Global Zone still sees one tree, with normal access
- GZ defines sub-tree for each LZ
- A process in an LZ only sees its own sub-tree
- All access is (still) controlled by the kernel
- NFS mount points are per-zone

DC Modifications

Users

- User name space is unique per zone
- Accounting
  - System V accounting generates records specific to that zone
  - Solaris extended accounting tools are zone-aware
    - GZ activity GZ leaves records in the GZ
    - LZ activity leaves records in that LZ and the GZ
  - Accounting controls can be configured per-zone
- Auditing
  - Solaris security auditing is zone-aware
  - Each zone can access its own audit records
  - GZ can access records of each zone separately or combined

DC Modifications

Communications: IPC

- IPC mechanisms that use shared files for communications can be used for inter-zone IPC
  - The GZ must configure the zones with shared access to an FS
- Sys V IPC that uses memory will work between processes in the same zone, but *not* between processes in different zones

DC Modifications

Communications: Networking

- Each zone can have an IP address – it must be unique
- LZ IP addr's can be multiplexed onto physical interfaces
- GZ can configure IPMP for physical interfaces
- GZ can configure IPSec and IPQoS per zone
- Each zone has its own TCP/UDP port ranges
- Use of ifconfig limited in an LZ
- Inter-zone: high bandwidth, low latency
- Broadcast and multicast work
- An LZ cannot change the routing table

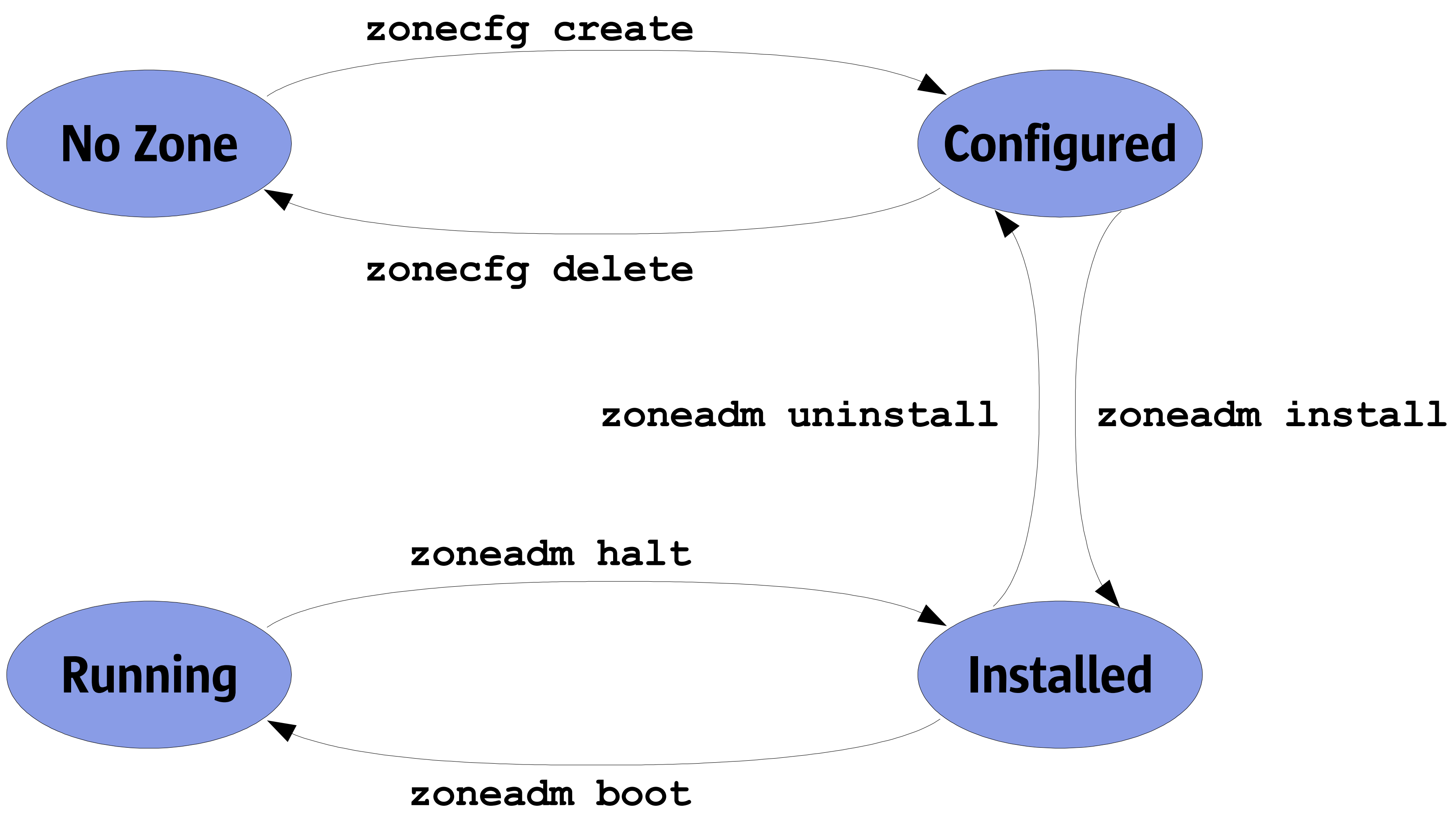
DC Modifications

Other

- Name space isolation for hosts, naming services
- Devices
  - /dev exists in an LZ, /devices does not
  - /dev includes a limited set, e.g. console, null, random
  - GZ can add devices to an LZ, but this is discouraged

Container Management

Lifecycle



$$\begin{tikzcd}
\text{No Zone} \arrow[rr, bend left, "\texttt{zonecfg create}"] & & \text{Configured} \arrow[ll, bend left, "\texttt{zonecfg delete}"] \arrow[d, bend left, "\texttt{zoneadm install}"] \\
\text{Running} \arrow[rr, bend left, "\texttt{zoneadm halt}"] & & \text{Installed} \arrow[ll, bend left, "\texttt{zoneadm boot}"] \arrow[u, bend left, "\texttt{zoneadm uninstall}"]
\end{tikzcd}$$

Container Management

Zone Creation

- GZ root runs "*zonecfg -z zonename -f cfgfile*"
    - **Sample zonecfg file at the end of presentation**
- Basic configuration parameters:
    - set zonepath=/zones/zone_roots/zone1
    - set autoboot=true
    - add net
        - set address=10.1.1.1
        - set physical=hme0
        - end
- GZ root runs "*zoneadm -z zonename install*"

Container Management

Boot Process

- GZ root runs "*zoneadm -z zonename boot*"
- Process
  - Kernel creates a zsched process for the zone
  - LZ sees zsched at root of process tree
  - zsched creates init which creates everything else
- Use "*zlogin -C zonename*" to watch
- Check status with: "*zoneadm list -cv*"

```
ID  NAME       STATUS      PATH
 0  global     running     /
 3  zone1      running     /zones/zone_roots/zone1
 5  zone2      installed   /zones/zone_roots/zone2
```

Container Management

Tearing Down...

- GZ root runs:
    - *zoneadm -z zonename halt*
    - *zoneadm -z zonename uninstall*
    - *zonecfg -z zonename delete*

Container Configuration
Adding Read-Only File Systems

- Must be configured by GZ-root with zonecfg
- FS's with installed packages are handled differently
- Basic configuration parameters for arbitrary filesystem:
  - add fs
    - set dir=/usr/local # LZ's path
    - set special=/opt/local # GZ's path
    - set type=lofs
    - end
- Basic configuration parameters for filesystem with packages:
  - add inherit-pkg-dir
    - set dir=/opt/sfw # Path in LZ and GZ
    - end

Container Configuration
Adding Read-Write File Systems

- Must be configured by GZ-root with zonecfg
- File system must be created by GZ-root
- Basic configuration parameters:
  - add fs
    - set dir=/mnt # LZ's (fixed) mount point
    - set special=/dev/dsk/c0t0d0s7
    - set raw=/dev/rdsk/c0t0d0s7
    - set type=ufs
    - end

Container Configuration

Direct Device Access

- Must be configured by GZ-root with zonecfg
- Consider potential security risks!
- Basic configuration parameters:
  - add device
    - set match=/dev/scsi/scanner/c3t4*
    - end

Container Configuration

Resource Management

- Must be configured by GZ-root with zonecfg
- Can be reconfigured dynamically
- Basic configuration parameters:
  - add rctl
    - set name=zone.cpu-shares
    - add value (priv=privileged,limit=20,action=none)
    - end

Container Management

Monitoring Commands

- These accept "*-z zonename*" and limit output or effects
  - ps, prstat, prstat, ptree, pkill, pgrep
- These accept "-Z" and add per-zone output
  - ps -Z
  - prstat -Z
  - df -Z
- In an LZ, netstat only shows your zone's network connections
- Pools and psets:
  - iostat, mpstat, prstat, and vmstat recognize pset limit
  - Same true for sar with some of its options

Container Management

Modified Commands

- ifconfig
  - From GZ: “*-zone*” places the interface in the GZ (default)
  - From GZ: “*zone zonename*” places the interface in zone zonename
  - From LZ: can only view per-zone info, cannot modify info
- *ppriv zone*: list all privileges available to the current zone
- *iostat* from LZ
  - shows info about the zone's NFS mounts
  - shows logical drive names (“sd5”) but not real device names (“c0t0d0” or “/devices/...”)

Container Management

Modified Commands

- uptime only shows information for your zone

Container Management

Other Limitations

- Attempts to gather info about processes in other zones result in an indication that the process doesn't exist
- Root in GZ must be careful with some methods, e.g. “*pkill sendmail*”
- CacheFS mounts not permitted in an LZ
- An LZ cannot be an NFS server
- An LZ cannot discover its IP address through DHCP (yet)
- An LZ can change its hostname, but not its zonename
- An LZ's default router must be in the same subnet as the LZ
- IPsec works, but IKE doesn't (yet)

Container Management

Other Limitations

- Some commands will not work in an LZ
  - prtdiag
  - prtconf (supplies some basic info)
  - eeprom
  - snoop
  - sysdef [-d] [-D] [-i]

Sample Uses

- Server consolidation
- Honeypot
- Multi-tier architecture, esp. combined with DTrace
- Group of applications that need a security boundary
- Group of applications that use a unique resource, e.g. port 80
- Utility Computing Model
- Test/POC environments: simulate a set of production systems

But beware of...

- Over-subscription (particularly memory)
- Unintended SPOFs

Sample Use

Web Server Consolidation

Before

- 40 web servers, 1 CPU each, 2U each - two racks
- Mix of Solaris, Linux, Windows
- Apache
- 5-10% CPU utilization (equivalent to 2-4 CPUs)

After

- Target: 50% CPU utilization
- Two V40z, 4x Opteron CPUs (4U each)
- Solaris 10, 20 zones per system

Sample Use

“Honeypot” - Using Observability to Monitor Security

Before

- Winnie the Pooh sticks his paw in the jar, notices that a bee (root) is logged in, wanders off in search of an unmonitored system

After

- A site that is regularly attacked installs Solaris 10
- Global zone uses one NIC attached to the network's DMZ
- A local zone uses a different NIC, attached to the external net
- A monitoring process in the GZ can watch the LZ
- A process in the LZ *cannot* detect the monitor

Sample zonecfg file

```
ambreesh@vitalstatistix:/Zones> more twilight.cfg
create
set zonepath=/Zones/twilight
set autoboot=false
add net
set physical=bge0
set address=192.168.100.11/24
end
add inherit-pkg-dir
set dir=/opt
end
add fs
set dir=/export/home
set type=lofs
add options [rw,nodevices]
set special=/export/home/twilight
end
add rctl
set name=zone.cpu-shares
add value(priv=privileged,limit=10,action=none)
end
verify
commit
ambreesh@vitalstatistix:/Zones>
```

Sample zone install

```
root@vitalstatistix:/Zones# zoneadm list -cv
  ID NAME             STATUS         PATH
   0 global           running        /
   - twilight         installed      /Zones/twilight
   - espn             installed      /Zones/espn
root@vitalstatistix:/Zones# zonecfg -z red -f red.cfg
root@vitalstatistix:/Zones# zoneadm list -cv
  ID NAME             STATUS         PATH
   0 global           running        /
   - twilight         installed      /Zones/twilight
   - espn             installed      /Zones/espn
   - red              configured     /Zones/red
root@vitalstatistix:/Zones# zoneadm -z red install
Preparing to install zone <red>.
Creating list of files to copy from the global zone.
Copying <2411> files to the zone.
Initializing zone product registry.
Determining zone package initialization order.
Preparing to initialize <1163> packages on the zone.
Initialized <1163> packages on zone.
Zone <red> is initialized.
Installation of these packages generated errors: <CSWisaexec SUNWzfs>
Installation of <2> packages was skipped.
Installation of these packages generated warnings: <SUNWsom SUNWauda SUNWxorg-server SUNWgnome-audio-share
        SFWkde>
The file </Zones/red/root/var/sadm/system/logs/install_log> contains a log of the zone installation.
root@vitalstatistix:/Zones# zoneadm -z red boot     <--- at this point, type zlogin -C red in a separate window -->
root@vitalstatistix:/Zones#
```

Thank You

Systems Software Products

Sun Microsystems, Inc.

SOLARIS™

Sun microsystems

We make the net work.