Best Practices for Sun Solaris Containers and VMware Infrastructure

Bob Netherton

Technical Specialist,
Solaris Adoption

VMWORLD 2006

Agenda

- What is OS Virtualization ?
- What are Solaris Containers and how do they work ?
- Is it “VMware ESX or Containers” or “VMware ESX and Containers” ?
- Examination of common use cases

Background

- Assumptions:
  - You already understand VMware Infrastructure components
  - You have heard of Solaris Containers or Zones
- Observations:
  - Solaris Containers and VMware Infrastructure (ESX) technologies are complementary
  - Each provides a unique set of capabilities and efficiencies that can be leveraged together
- The key to success is knowing when to use each technology

Solaris Containers and OS Virtualization

- Multiple isolated execution environments within one Solaris instance
- Includes resource management, security, failure isolation
- Lightweight, flexible, efficient
    - More than 8,000 zones per system (or dynamic system domain)
- One operating system to manage
    - Device configuration details hidden
- Components:
    - Workload identification & accounting, process aggregation
    - Resource management (CPU, memory, ...)
    - Security/namespace isolation (zones)
- Features can be used separately or in combination

Evolution of Solaris Containers

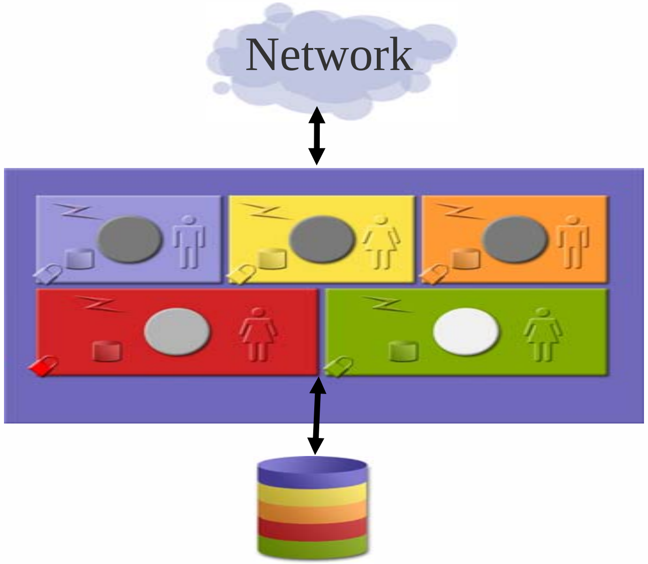
- **Solaris Containers prior to Solaris 10**
  - Introduced in Solaris 2.6 as SRM 1.0 [aka “Share II” scheduler]
  - Integrated into Solaris 9; new commands
  - Redesigned Fair Share Scheduler
  - Resource Capping Daemon
  - Introduced Extended Accounting
  - Better integration with Processor Pools/Sets
- **New In Solaris 10:**
  - Partitioning and Isolation with Zones
  - Dynamic Control of Pools
  - More Dynamic Resource Controls
    - Trend is to move away from /etc/system

Solaris Containers Components

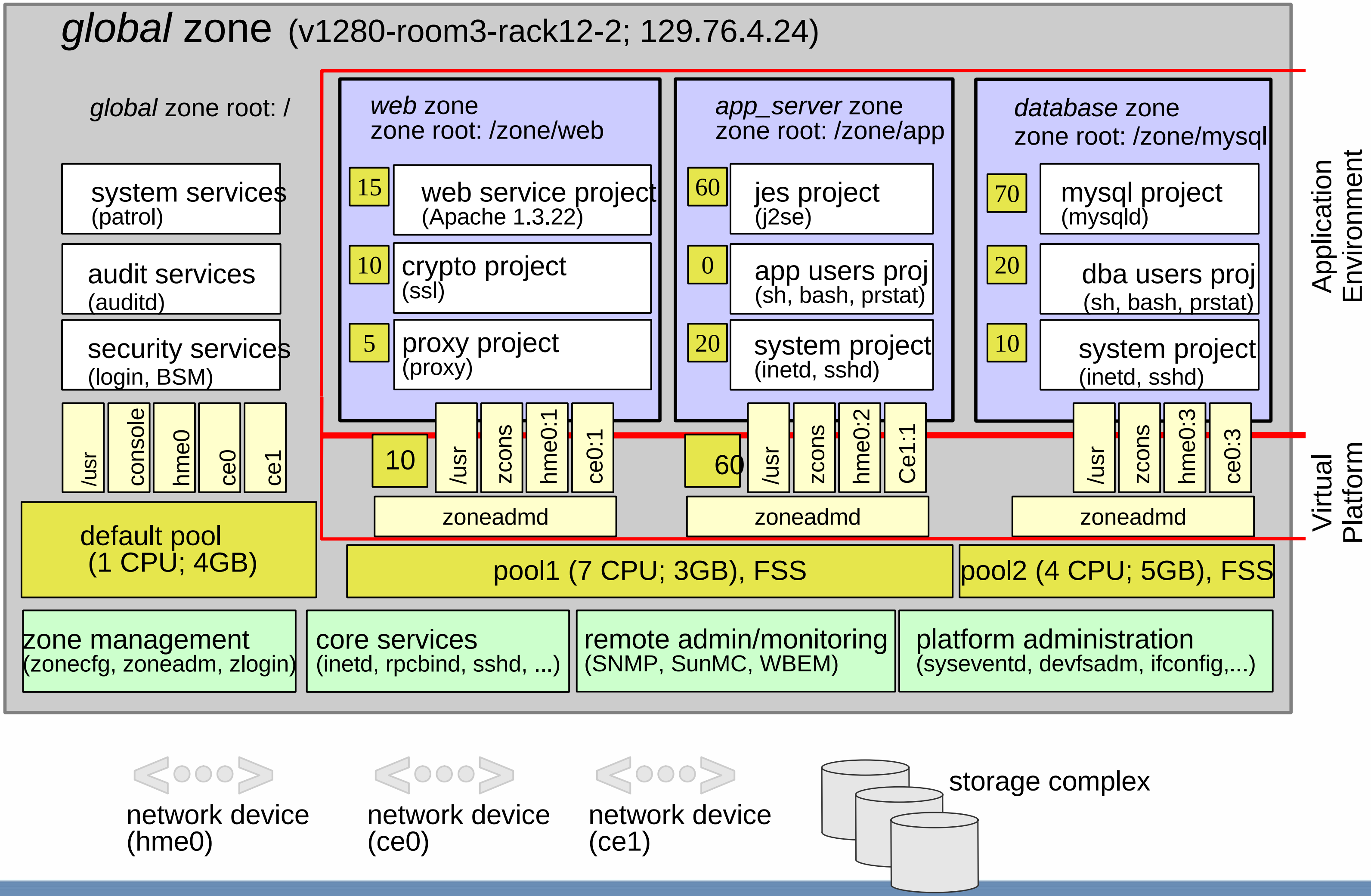
- Workload identification
  - Process aggregation via tasks, projects
  - Resource usage log with extended accounting
- Resource Management Tools
  - Guarantee minimum CPU use (FSS)
  - Limit maximum CPU use (pools, processor sets)
  - Limit physical memory use (resource capping daemon)
  - Limit virtual memory use (projects)
  - Limit network bandwidth use (ipqos)
- Workload isolation features
  - Privileges
  - Zones

OS Virtualization through Solaris Zones

- Virtualizes OS layer: file system, devices, network, processes
- Provides:
  - Privacy: can't see outside zone
  - Security: can't affect activity outside zone
  - Failure isolation: application or service failure in one zone doesn't affect others
- Lightweight, granular, efficient
- Complements resource management
- No porting; ABI/APIs are the same
- Requires no special hardware assist



Example



global zone (v1280-room3-rack12-2; 129.76.4.24)

global zone root: /

system services (patrol)

audit services (auditd)

security services (login, BSM)

/usr | console | hme0 | ce0 | ce1

default pool (1 CPU; 4GB)

web zone
zone root: /zone/web

15 web service project (Apache 1.3.22)

10 crypto project (ssl)

5 proxy project (proxy)

10 /usr | zcons | hme0:1 | ce0:1

zoneadmd

app_server zone
zone root: /zone/app

60 jes project (j2se)

0 app users proj (sh, bash, prstat)

20 system project (inetd, sshd)

60 /usr | zcons | hme0:2 | Ce1:1

zoneadmd

database zone
zone root: /zone/mysql

70 mysql project (mysqld)

20 dba users proj (sh, bash, prstat)

10 system project (inetd, sshd)

/usr | zcons | hme0:3 | ce0:3

zoneadmd

Application Environment

Virtual Platform

pool1 (7 CPU; 3GB), FSS

pool2 (4 CPU; 5GB), FSS

zone management (zonecfg, zoneadm, zlogin)

core services (inetd, rpcbind, sshd, ...)

remote admin/monitoring (SNMP, SunMC, WBEM)

platform administration (syseventd, devfsadm, ifconfig,...)

network device (hme0)

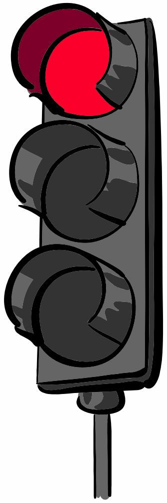
network device (ce0)

network device (ce1)

storage complex

Solaris Security

- User Rights Management
  - Limit access to privileged commands and operations
  - Manage “who can do what” centrally
  - Audit and report privileged command use
- Process Rights Management
  - Grant or revoke fine-grained privileges to individual processes and applications
  - Implement “Least Privilege”
    - Applications can only do exactly what they require to operate
  - Usually removes the need to run as root



Solaris Security (cont)

- More than 40 specific rights historically associated with UID 0 (root)
- For legacy compatibility, UID 0 has all rights by default
- Basic users have very few default rights
- Selectable privilege inheritance
- Role-based access control framework enables:
  - Privileges to be assigned to a role
  - Specific users to temporarily take on a role, gaining its privileges
- Kernel enforces rules based on uid and current privileges
  - No more “if (uid==0)”

Solaris Zones and Security

- Each zone has a security boundary
- Runs with subset of privileges(5)
- A compromised zone cannot escalate its privileges
- Important name spaces are isolated
- Processes running in a zone are unable to affect activity in other zones
- Zone-aware audit:
  - Global zone administrator can specify whether auditing should be global or per-zone
  - If per-zone, each zone administrator can configure and process their audit trails independently
- Solaris 10 11/06 introduces configurable privileges

Solaris Zones Security Limits

contract_event	Request reliable delivery of events	proc_lock_memory	Lock pages in physical memory
contract_observer	Observe contract events for other users	proc_owner	See/modify other process states
cpc_cpu	Access to per-CPU perf counters	proc_priocntl	Increase priority/sched class
dtrace_kernel	DTrace kernel tracing	proc_session	Signal/trace other session process
dtrace_proc	DTrace process-level tracing	proc_setid	Set process UID
dtrace_user	DTrace user-level tracing	proc_taskid	Assign new task ID
file_chown	Change file's owner/group IDs	proc_zone	Signal/trace processes in other zones
file_chown_self	Give away (chown) files	sys_acct	Manage accounting system (acct)
file_dac_execute	Override file's execute perms	sys_admin	System admin tasks (e.g. domain name)
file_dac_read	Override file's read perms	sys_audit	Control audit system
file_dac_search	Override dir's search perms	sys_config	Manage swap
file_dac_write	Override (non-root) file's write perms	sys_devices	Override device restricts (exclusive)
file_link_any	Create hard links to diff uid files	sys_ipc_config	Increase IPC queue
file_owner	Non-owner can do misc owner ops	sys_linkdir	Link/unlink directories
file_setid	Set uid/gid (non-root) to diff id	sys_mount	Filesystem admin (mount,quota)
ipc_dac_read	Override read on IPC, Shared Mem perms	sys_net_config	Config net interfaces,routes,stack
ipc_dac_write	Override write on IPC, Shared Mem perms	sys_nfs	Bind NFS ports and use syscalls
ipc_owner	Override set perms/owner on IPC	sys_res_config	Admin processor sets, res pools
net_icmpaccess	Send/Receive ICMP packets	sys_resource	Modify res limits (rlimit)
net_privaddr	Bind to privilege port (<1023+extras)	"ys_suser_compat	3rd party modules use of suser
net_rawaccess	Raw access to IP	sys_time	Change system time
proc_audit	Generate audit records		
proc_chroot	Change root (chroot)		
proc_clock_highres	Allow use of hi-res timers		
proc_exec	Allow use of execve()	Interesting	Some interesting privileges
proc_fork	Allow use of fork*() calls	Basic	Non-root privileges
proc_info	Examine /proc of other processes	Removed	Not available in Zones

Processes

- Certain system calls are not permitted or have restricted scope inside a zone
- From the global zone, all processes can be seen but control is privileged
- From within a zone, only processes in the same zone can be seen or affected
- `proc(4)` has been virtualized to only show processes in the same zone

```
# prstat -Z
   PID USERNAME  SIZE   RSS STATE  PRI NICE      TIME  CPU PROCESS/NLWP
  1344 root     8956K 8108K sleep   59    0   0:00:04 2.0% svc.configd/14
  1342 root     7312K 6456K sleep   59    0   0:00:01 0.4% svc.startd/12
  1460 root     3824K 2932K sleep   59    0   0:00:00 0.1% inetd/4


ZONEID    NPROC  SIZE   RSS MEMORY      TIME  CPU ZONE
     1       23   78M   46M   4.5%   0:00:05 2.8% zone1
```

Networking and Interprocess Communication

- Single TCP/IP stack for the system (today) so that zones can be shielded from configuration details for devices, routing and IPMP
- Each zone can be assigned IPv4/IPv6 addresses and has its own port space
- Applications can bind to `INADDR_ANY` and will only get traffic for that zone
- Zones cannot see the traffic of others
- Global zone can snoop traffic of all zones
- Expected IPC mechanisms such as System V IPC, STREAMS, sockets, `libdoor(3LIB)` and loopback transports are available inside a zone
- Key name spaces virtualized per zone
- Inter-zone communication is available using standard network interfaces over a private memory channel.
- Global zone can setup rendezvous too, although this is not commonly needed

Devices and Filesystems

- Unlike `chroot(2)`, processes cannot escape out of a zone's filesystems
- Additional directories can be mounted read-write
  - **Example `/usr/local`**
- Filesystems mounted by zoneadmd at zone boot time.
- Global zone managed filesystems also supported
  - Third party filesystems also work (ex: VxFS)
- Zones see a subset of “safe” pseudo devices in their **`/dev`** directory
  - Devices like **`/dev/random`** are safe but others like **`/dev/ip`** are not
- Zones can modify the permissions of their devices but cannot `mknod(2)`
- Physical device files like those for raw disks can be put in a zone with caution
  - Often unnecessary due to on-disk filesystem support in zonecfg

Zones and Solaris Dynamic Tracing (DTrace)

- Zonename variable available
- Example: Count syscalls by zone:

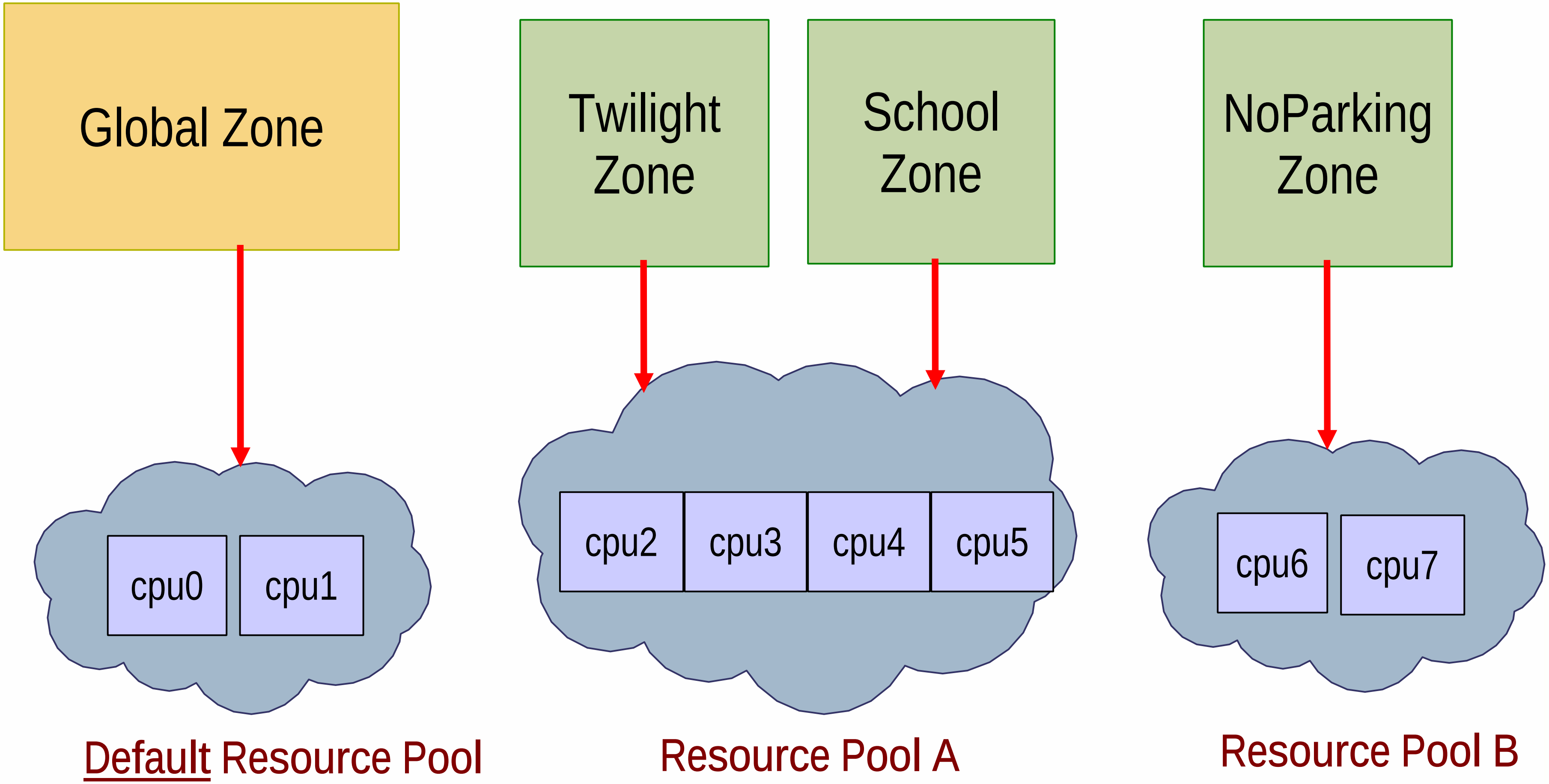
```
# dtrace -n 'syscall:::/zonename==”red”/
   {@[probefunc=count()}'
```

- Also available: `curpsinfo->pr_zoneid`
- DTrace can be useful for tracing multiple application tiers in conjunction with zones
  - Eliminates complexities such as clock skew
- Solaris 10 11/06 configurable privileges will allow dtrace_user and dtrace_proc to be granted to a zone
  - Allows tracing of processes (pid) and system calls (syscall)

Zones, Resources and Limits

- By default, all zones use all CPUs
  - Also, tools like prstat base %'s on all CPUs
- Restricted view is enabled automatically when resource pools are enabled
  - virtualized view based on the pool (pset) binding
  - Affects `iostat(1M)`, `mpstat(1M)`, `prstat(1M)`, `psrinfo(1M)`, `sar(1)`, etc.
  - `sysconf(3C)` (when detecting number of processors) and `getloadavg(3C)`
  - numerous `kstat(3KSTAT)` values from the `cpu`, `cpu_info` and `cpu_stat` publishers
- Oracle licensing to pool size

Resource Pools



Global Zone

Twilight Zone

School Zone

NoParking Zone

cpu0

cpu1

cpu2

cpu3

cpu4

cpu5

cpu6

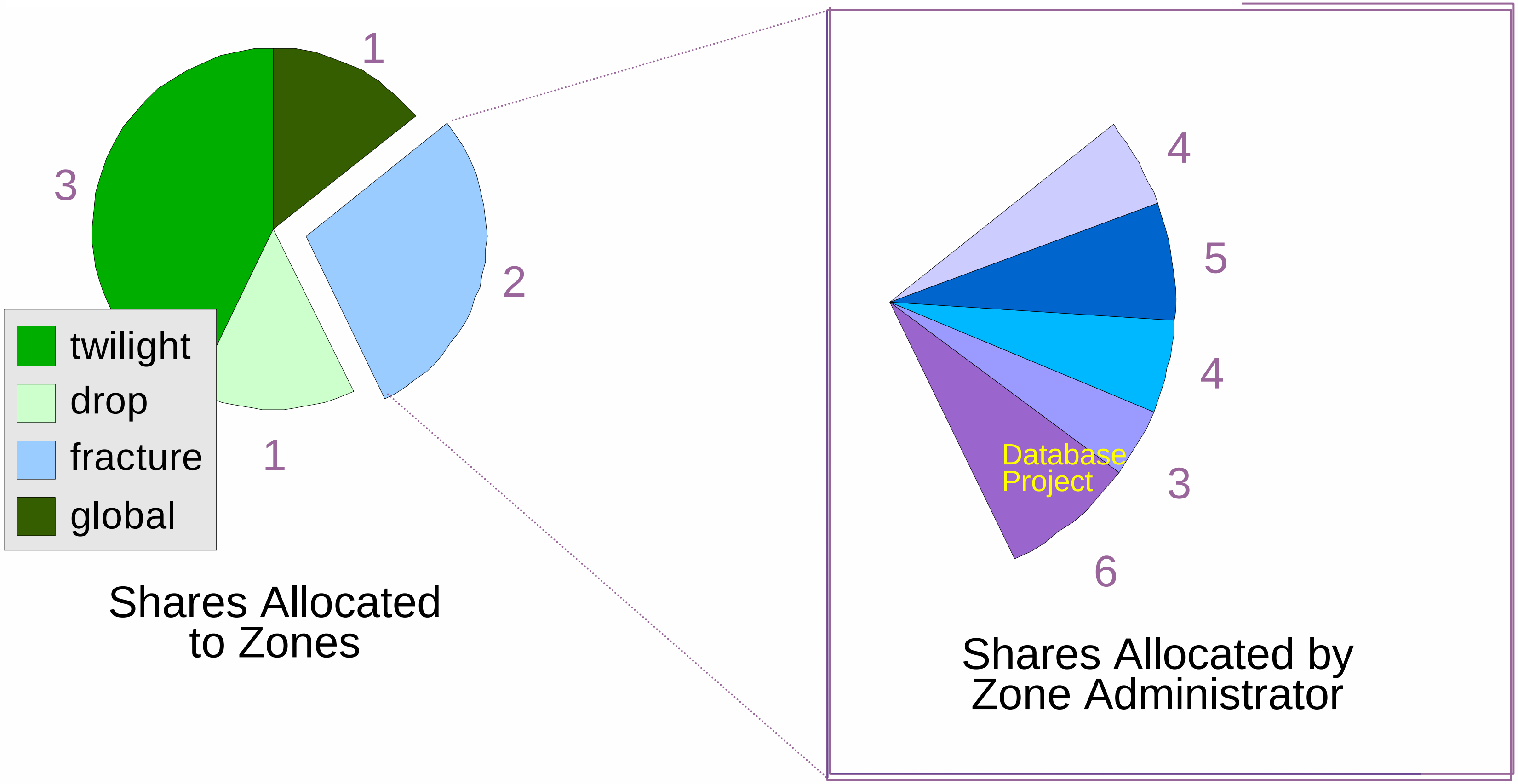
cpu7

Default Resource Pool

Resource Pool A

Resource Pool B

Zones and the Fair Share Scheduler (FSS)



Shares Allocated to Zones

Shares Allocated by Zone Administrator

$$\frac{2}{(3+1+2+1)} \times \frac{6}{(4+5+4+3+6)} = \frac{2}{7} \times \frac{6}{22} = \frac{6}{77} \sim 7.8\%$$

Sparse vs Whole Root Zones

- Each zone is assigned its own root file system and cannot see that of others
- The default file system configuration is called a “sparse-root” zone
  - The zone contains its own writable /etc, /var, /proc, /dev
  - Inherited file systems (/usr, /lib, /platform, /sbin) are read-only mounted via a loopback file system (LOFS)
  - /opt is a good candidate for inheriting
- A zone can be created as a “whole-root” zone
  - The zone gets its own writable copy of all Solaris file systems
- Advantages of a sparse root zone
  - Faster patching and installation due to inheritance of /usr and /lib
  - Read-only access prevents trojan horse attacks against other zones
  - Libraries shared across all zones reducing VM footprint

Packages and Patches

- Zones can add and remove own packages and patches (i.e. database)
  - Assuming packages don't conflict with global zone packages (or allzone packages)
- System Patches
  - Applied in global zone
  - Then in each non-global zones (zone will automatically boot -s to apply patch)
- Package types
  - `SUNW_PKG_HOLLOW:` Package info exists (to satisfy dependencies) but its contents are not present.
  - `SUNW_PKG_ALLZONES:` Package will be kept consistent between the global zone and all non-global zones (e.g. kernel drivers).
  - `SUNW_PKG_THISZONE:` If true, package installs only in the current zone (like pkgadd -G). If installed in the global zone, it will not be made available to future zones.

Zone Administration

- `zonecfg(1M)` is used to specify resources (e.g. IP interfaces) and properties (e.g. resource pool binding)
- `zoneadm(1M)` is used to perform administrative steps for a zone such as list, install, (re)boot, halt
- Installation creates a root file system with factory-default editable files
- A zone can be cloned very quickly using ZFS
- A zone can be moved to another system with detach/attach
- `zlogin(1)` is used to access a zone
  - `zlogin -C` to access the zone console

Zone Installation Process

- By default, all of the files that are packaged in the global zone are stored in the new zone
- Packaged files are copied directly out of the global zone's root file system except for those that are editable or volatile (see pkgmap(4))
- Editable and volatile files are copied from the sparse-root package archive
  - holds factory default copies of files
- A properly configured sparse-root zone is typically about 70-100MB; a whole-root zone is 3-5GB depending on installed packages

When to Use a Whole Root Zone

- Use full root zones when writes into /usr or /lib cannot be contained
  - Writable loopback mounts for individual directories (such as /usr/java) can be used for sparse root zones
  - Sometimes this is not practical (example: /usr/bin)
  - Use of writable loopback mounts makes /opt a good candidate for inheritance
- Requirement to patch Solaris user components individually
  - Third party software typically installed in /opt
- Use a sparse root zone for all other situations

Single or Multiple Applications Zones

- Single application zones
  - Low overhead (administrative and performance) makes this a recommended practice
  - All configuration files are in the default location
  - Virtualized IP space allows applications to reside on well known ports
  - Patching is simplified due to applications being where they are expected
- Multiple application zones
  - When applications require or can benefit from shared memory

VMware and Solaris Containers General Approach

- Use VMware when
  - Using heterogeneous or multiple (incompatible) versions of operating systems
  - Consolidated privileged applications are unstable
  - Operating system maintenance windows become unmanageable
  - Requiring live migration
  - Running obsolete operating systems on current hardware
- Use Solaris Containers when
  - Fine grained control of resource limits
  - Leveraging advanced Solaris features such as DTrace, Fault Management (FMA), ZFS
  - Resource sharing between environments can reduce platform costs
  - Deploying extremely heavy or very light services
    - Applications require high I/O throughput (databases)
- Combine generously as real world conditions are never sim

Solaris Containers Best Practices

- Use sparse root zones where possible
  - Maximize sharing of components
  - Minimize memory footprint (shared libs, binaries)
- Use full root zones only where needed
  - Extensive writing into /usr
  - Core component patch testing
  - Use of ZFS clones will make this much more attractive
- Group applications into zones
  - By shared memory requirements
  - By user credential domain
- Use loopback file system mounts to share data
- Use NFS to share data for zones that will be migrated

Solaris Containers Best Practices (cont)

- File system backup can be run in the global zone
  - Non-global zones have no private file system data that is not visible to the global zone
- Run backup clients in the non-global zone when there is some application state that needs to be captured or modified
- Run a minimum number of services in the global zone
  - ssh
  - Intrusion detection and auditing
  - Hardware monitoring
  - Accounting
  - Backup

Use Case 1: You are in a maze of web servers, all alike

- Considerations
  - Web servers prefer to live on well known ports
  - Server utilization can be very low
  - Many configurations are basic
  - Consolidating in a single operating system can become very complex
  - Classic partitioning problem in disguise
- Recommendation
  - One web server instance per Solaris zone
  - Very few operating system dependencies
  - Configuration files are all in their well known location
  - Patch automation is simplified
  - Separate content creation for a more secure solution
  - Can leverage Solaris least privilege

Use Case 2: Web 2.0

- Considerations
  - Operating system dependencies more complicated
  - Avoid unintended application linkages that make future updates or redeployment difficult
  - Leverage operating system hardening and privilege minimization
  - Require fine grain control over resource utilization
- Recommendation
  - Use Solaris zones with one application instance per zone
  - Deploy only the OS components necessary to support the service
  - Use configurable privileges to limit access to memory, network interfaces and kernel modules.
- Exceptions
  - Services that have dependencies on kernel modules
  - Heterogeneous operating system requirements

Use Case 3: ERP in a box

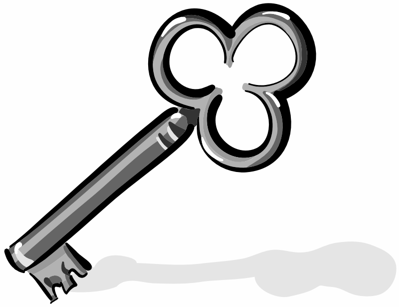
- Considerations
  - Different than typical ERP landscape
    - Trade off database performance considerations for reduced footprint
  - Not all features or applications run on all operating systems
  - Will require a combination of virtualization and partitioning
  - Desire fine grain control of resources
  - Observability and security are desired features, especially in development
- Recommendation
  - Use VMware ESX server to host multiple operating systems
  - Run database in one zone and application logic in separate zones based on software scalability features
    - Solaris Dynamic Tracing can be used across tiers
  - Host additional guest virtual machines for interfaces and application features not available on Solaris

Use Case 4: Enterprise Java Application Development

- Considerations
  - Leverage advanced development tools such as DTrace
    - Java Virtual Machine DTrace provider is very handy
  - Isolate to minimize impact on other developers
  - Develop in same environment as deployment
  - Rapidly provision complete software stacks
- Recommendation
  - Create development zones that mirror production and test environments
  - Use zone privilege limits to safely delegate administrative roles to developers
- Exceptions
  - Heterogeneous platform development
  - Develop on multiple operating system versions

Conclusion

- Solaris Containers and VMware Infrastructure (ESX) technologies are complementary
    - Each provides a unique set of capabilities and efficiencies that can be leveraged together
- The key to success is knowing when to use each technology



References and Additional Reading

- Zones BigAdmin site:
  - http://www.sun.com/bigadmin/content/zones
- *Solaris Zones: Operating System Support for Server Consolidation. (LISA 2004, available from BigAdmin)*
- *Solaris Containers Blueprint:*
  - *http://www.sun.com/blueprints/0505/819-2679.html*
- Solaris Kernel Engineering and Field Technical Weblogs
  - http://blogs.sun.com/comay
  - http://blogs.sun.com/dp
  - http://blogs.sun.com/jclingan
  - http://blogs.sun.com/joostp
- Zones/Containers FAQ on opensolaris.org
- zones-interest@opensolaris.org mailing list
- Solaris 10 global zone with 3 containers (web, app, and dba)
  - http://www.vmware.com/vmtn/appliances/directory/227

Thank you!

Presentation Download

Please remember to complete your
session evaluation form
and return it to the room monitors
as you exit the session

The presentation for this session can be downloaded at
http://www.vmware.com/vmtn/vmworld/sessions/

Enter the following to download (case-sensitive):

Username: cbv_rep
Password: cbvfor9v9r

VMWORLD 2006