Utility Tools for Management of Solaris™ Containers

September 2005
Lei Liu
Sun Microsystems

Introduction

The Solaris 10 Operating System introduced zone-oriented virtualization and partition models on the Solaris platform. These models require zone-specific performance measurements to instrument zones at the system level. Moreover, applications and services are hosted within the isolated environment provided by the zones. How these applications perform needs to be inspected at the zone level. The Solaris Zones partitioning technology, a component of Solaris Containers, introduces a new performance metric to the traditional Solaris OS. Another component of the Solaris Containers environment, resource management, enables you to control how applications use available system resources. In the meantime, resource management monitors how computing resources are allocated to processes.

This document is designed to help management service providers, management solution providers, architects, data center administrators, and performance engineers to utilize the Solaris platform's built-in utility tools to discover and instrument zones. The resource management feature and extended accounting facility for zone monitoring are also recommended. When these utilities are integrated with management entities, management applications can monitor and set alarms for zone status on the Solaris platform with minimal coding involved.

Background

Business services are designed to satisfy service levels with capacity growth planning to handle peak workloads and to achieve revenue growth. However, system resources are not fully utilized during normal workloads. Resources need to be dynamically allocated to business-critical services and applications while ensuring that services with lower priorities will not be compromised. Therefore, to reduce the cost of managing vast networks of servers with software components installed on thousands of nodes, IT managers are shifting their focus from increasing performance and availability to reducing costs of IT infrastructure and improving end-user service levels. Server, application, and service consolidation is one of the important areas that requires system vendor support at the system level.

Businesses can accomplish this consolidation through server virtualization. Server virtualization allows data centers to be visualized and managed as a fabric of interconnected computing resources rather than as a room filled with individual systems. Solaris Containers are designed and implemented as a system-level virtualization model to provide a means of virtualizing the operating system environment within a shared instance of the Solaris OS. Solaris Containers offer a complex execution environment for a set of software services and a separate Solaris environment within a single Solaris instance.

Solaris Zones offer virtual mapping from software services to platform resources and allow application components to be isolated from each other. Hence, zones create an isolated environment for running applications. This isolation prevents processes that are running in one zone from monitoring or affecting processes running in other zones. Zones are ideal for environments that consolidate a number of applications on a single server. The cost and complexity of managing numerous machines make it advantageous to consolidate several applications on larger, more scalable servers.

Solaris Zones provide the following functions:

- **Security**
  - Network services can be run in a zone, limiting the risks that can affect systems and other zones in case of a security violation.

- **Isolation**
  - Applications requiring exclusive access to global resources can run on the same systems using zones.

- **Virtualization**
  - A virtualization environment is presented to applications, which abstracts the underlying physical implementation of hardware and software.

- **Granularity**
  - Sub-CPU granularity: As a result, zones do not require dedicated resources such as CPU and memory. A single system can host many zones.

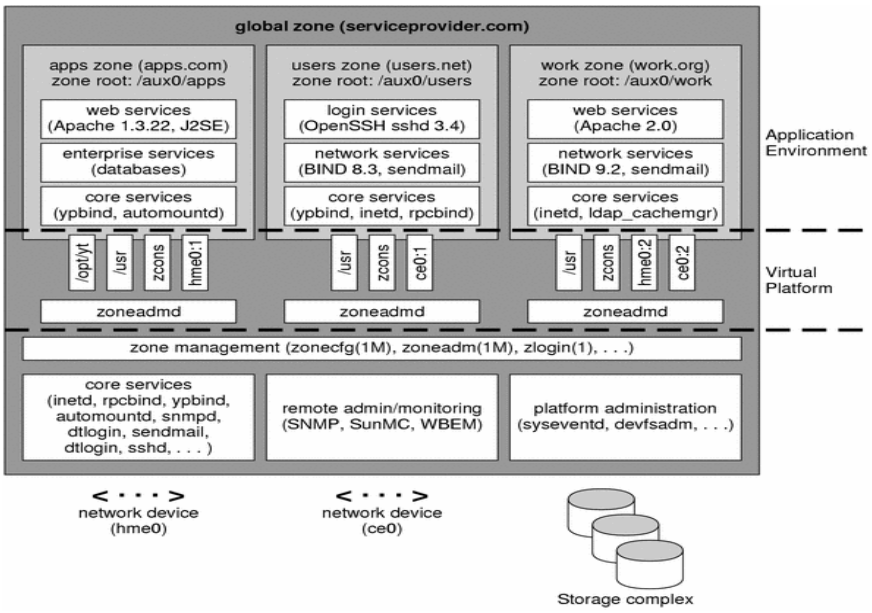

Figure 1: Context of Solaris Zones Feature

Solution Description

The Solaris OS has built in a series of utility tools to discover zones and to instrument a zone-specific matrix. The utility can be easily integrated with a management entity on the managed node for system and performance management solution providers. The following sections walk through the flow of zone management to illustrate the usage of the utilities. In addition, by leveraging the rlogin and ssh functionality, the management host can remotely instrument the managed hosts.

Zone Service Discovery

The zones facility in the Solaris OS provides an isolated environment for running applications. Processes running in a zone are prevented from monitoring or interfering with other activity in the system. Access to other processes, network interfaces, file systems, devices, and interprocess communication facilities is restricted to prevent interaction between processes in different zones. The `zones(5)` service is managed by the service management facility, `smf(5)`, under the service identifier `svc:/system/zones:default`. The service's status can be queried using the `svcs(1)` command. For more information see:

- `zones(5)`: http://docs.sun.com/app/docs/doc/816-5175/6mbba7f4t?a=view
- `smf(5)`: http://docs.sun.com/app/docs/doc/816-5175/6mbba7f3n?a=view
- `svcs(1)`: http://docs.sun.com/app/docs/doc/816-5165/6mbb0m9th?a=view

The `svcs(1)` command displays information about service instances as recorded in the service configuration repository.

```
svcs

STATE STIME FMRI

online Jun_17 svc:/system/zones:default
```

This lists the zone service instances:

```
svcs -o state,nstate,fmri zones

STATE NSTATE FMRI

online - svc:/system/zones:default
```

This lists detailed information about all instances of `system/zones:`

```
svcs -l system/zones

fmri         svc:/system/zones:default

name         Solaris zones

enabled      true

state        online

next_state   none

state_time   Fri Jun 17 08:16:40 2005

logfile      /var/svc/log/system-zones:default.log

restarter    svc:/system/svc/restarter:default

dependency   require_all/none svc:/milestone/multi-user-server (online)
```

Zone Instance Discovery

As an application container that is maintained by the operating system runtime, a zone can be discovered by the `zoneadm(1M)` utility. The Solaris utility tool `zoneadm(1M)` can be used to display the name of the current zones, or the specified zone if indicated. By default, all running zones are listed.

To discover all installed zones, you can use the `zoneadm(1M)` command:

```
zoneadm list -i
```

Execute the following command to display all configured zones and display verbose information, including zone name, ID, current state, root directory, and options.

```
zoneadm list -cv

ID     NAME         STATUS       PATH

0      global       running      /

5      webzone      running      /zone1

6      dbzone       running      /zone2

8      midtierzone  running      /zone3
```

The valid zone states are categorized in the following table.

State	Description
CONFIGURED	Indicates that the configuration for the zone has been completely specified and committed to stable storage.
INCOMPLETE	Indicates that the zone is in the midst of being installed or uninstalled, or was interrupted in the midst of such a transition.
INSTALLED	Indicates that the zone's configuration has been instantiated on the system: packages have been installed under the zone's root path.
READY	Indicates that the "virtual platform" for the zone has been established. Network interfaces have been plumbed, file systems have been mounted, devices have been configured, but no processes associated with the zone have been started.
RUNNING	Indicates that user processes associated with the zone application environment are running.
SHUTTING_DOWN DOWN	Indicates that the zone is being halted. The zone can become stuck in one of these states if it is unable to tear down the application environment state (such as mounted file systems) or if some portion of the virtual platform cannot be destroyed. Such cases require operator intervention.

Table 1: Zone State Categorization

To list all the running zones, `zoneadm(1M)` with the subcommand `list` can be used:

```
zoneadm list

global

webzone

dbzone

midtierzone
```

To find a specific zone regardless of its state, execute the following command. If you use this subcommand with the `zoneadm -z zonename` option, it lists only the specified zone, regardless of its state:

```
zoneadm -z <zonename> list -p
```

To adjust the `zoneadm(1M)` output as machine-parsable format, use the `-p` option in the `list` subcommand. If you want to know the current zone name only, you can utilize a user command `zonename(1)`.

Monitoring Zone Instances

This section addresses the zone-specific performance monitoring.

User Count Statistics

The `who(1)` utility can list the user's name, terminal line, login time, elapsed time since activity occurred on the line, and the process-ID of the command interpreter for each current Solaris system user:

```
zlogin <zonename> who | wc -l
```

Name	Description
`name`	User's login name.
`state`	Capability of writing to the terminal.
`line`	Name of the line found in `/dev`.
`time`	Time since user's login.
`idle`	Time elapsed since the user's last activity.
`pid`	User's process ID.
`comment`	Comment line in `inittab(4)` (see: http://docs.sun.com/app/docs/doc/816-5174/6mbb98ug1?a=view).
`exit`	Exit status for dead processes.

Table 2: `who(1)` General Format

Zone Statistics Report

In addition, the `prstat(1M)` utility iteratively examines all active processes on a specific zone and reports statistics based on the selected output mode and sort order. The `prstat (1M)` utility provides options to examine only processes matching specified PIDs, UIDs, zone IDs, CPU IDs, and processor set IDs:

```
prstat -v -z <zonename>

Report active process statistics
```

To report the five most active processes, use:

```
prstat -z <zonename>
```

Name	Description
`PID`	The process ID of the process.
`USERNAME`	The real user (login) name or real user ID.
`SIZE`	The total virtual memory size of the process, including all mapped files and devices, in kilobytes (`K`), megabytes (`M`), or gigabytes (`G`).

Name	Description
RSS	The resident set size of the process (RSS), in kilobytes (K), megabytes (M), or gigabytes (G). The RSS value is an estimate provided by proc(4) (http://docs.sun.com/app/docs/doc/816-5174/6mbb98uim?a=view) that might underestimate the actual resident set size. Users who want more accurate usage information for capacity planning should use the -x option to pmap(1) instead (see: http://docs.sun.com/app/docs/doc/816-5165/6mbb0m9on?a=view).
STATE	The state of the process: cpuN Process is running on CPU N. Sleeping: Process is waiting for an event to complete. Runnable: Process in run queue. Zombie state: Process terminated and parent not waiting. Stop: Process is stopped.
PRI	The priority of the process. Larger numbers mean higher priority.
NICE	Nice value used in priority computation. Only processes in certain scheduling classes have a nice value.
TIME	The cumulative execution time for the process.
CPU	The percentage of recent CPU time used by the process. If executing in a non-global zone and the pools facility is active, the percentage will be that of the processors in the processor set in use by the pool to which the zone is bound.
PROCESS	The name of the process (name of executed file).
LWPID	The lwp ID of the lwp being reported.
NLWP	The number of lwps in the process.
USR	The percentage of time the process has spent in user mode.
SYS	The percentage of time the process has spent in system mode.
TRP	The percentage of time the process has spent in processing system traps.
TFL	The percentage of time the process has spent processing text page faults.
DFL	The percentage of time the process has spent processing data page faults.
LCK	The percentage of time the process has spent waiting for user locks.
SLP	The percentage of time the process has spent sleeping.
LAT	The percentage of time the process has spent waiting for CPU.
VCX	The number of voluntary context switches.
ICX	The number of involuntary context switches.
SCL	The number of system calls.
SIG	The number of signals received.

Table 3: prstat(1M) General Format

Note: The snapshot of system usage displayed by prstat(1M) is true only for a split second, and it may not be accurate by the time it is displayed. When the -m option is specified, prstat(1M) tries to turn on microstate accounting for each process; the original state is restored when prstat(1M) exits.

Sample output looks like this:

```
prstat -z <zonename> -n 5

PID USERNAME SIZE  RSS   STATE PRI NICE TIME   CPU PROCESS/NLWP

7854   root   1240K 1040K sleep 39  0   0:00:00 0.0% sh/1

7855   root   4432K 4112K cpu17 49  0   0:00:00 0.0% prstat/1

19716  root   9608K 8856K sleep 59  0   0:00:40 0.0% svc.configd/13

19712  root   2360K 1664K sleep 59  0   0:00:08 0.0% init/1

19746  root   4288K 3440K sleep 59  0   0:01:54 0.0% nscd/24

Total: 32 processes, 101 lwps, load averages: 0.03, 0.02, 0.02
```

Hence, the zone process count can be monitored as:

```
prstat -z <zonename> | grep Total
```

In addition, CPU usage per process can be found with 5-minute intervals:

```
prstat -z <zonename> 300
```

The following command reports the average CPU workload over the last 15 minutes:

```
prstat -z <zonename> 900 | grep 'load averages'
```

A General System Activity Reporter

The `sar(1)` utility samples cumulative activity counters in the operating system.

Column	Description
`pread/s, pwrit/s`	Transfers using raw (physical) device mechanism. If run in a non-global zone and the pools facility is active, these values reflect activity on the processors of the processor set of the pool to which the zone is bound.
`rchar/s, wchar/s`	Characters transferred by read and write system calls. No incoming or outgoing `exec(2)` (see http://docs.sun.com/app/docs/doc/816-5167/6mbb2jafk?a=view) and `fork(2)` (see http://docs.sun.com/app/docs/doc/816-5167/6mbb2jag7?a=view) calls are reported. If run in a non-global zone and the pools facility is active, these values reflect activity on the processors of the processor set of the pool to which the zone is bound.

Column	Description
%ufs_ipf	The percentage of UFS inodes taken off the freelist by iget, which had reusable pages associated with them. These pages are flushed and cannot be reclaimed by processes. Thus, this is the percentage of igets with page flushes. If run in a non-global zone and the pools facility is active, these values reflect activity on the processors of the processor set of the pool to which the zone is bound.
msg/s, sema/s	Primitives per second. If run in a non-global zone and the pools facility is active, these values reflect activity on the processors of the processor set of the pool to which the zone is bound.
slock/s	Faults per second caused by software lock requests requiring physical I/O. If run in a non-global zone and the pools facility is active, these values reflect activity on the processors of the processor set of the pool to which the zone is bound.
%usr, %sys, %wio, %idle	Portion of time running in user mode, running in system mode, idle with some process waiting for block I/O, and otherwise idle. If run in a non-global zone and the pools facility is active, these values reflect activity on the processors of the processor set of the pool to which the zone is bound.
pswch/s	Process switches. If run in a non-global zone and the pools facility is active, these values reflect activity on the processors of the processor set of the pool to which the zone is bound.
rcvin/s, xmtin/s, mdmin/s	Receive, transmit, and modem interrupt rates. If run in a non-global zone and the pools facility is active, these values reflect activity on the processors of the processor set of the pool to which the zone is bound.

Table 4: sar(1) General Format

Zone CPU Monitoring

The mpstat(1M) utility reports processor statistics in tabular form. Each row of the table represents the activity of one processor. The first table summarizes all activity since boot. Each subsequent table summarizes activity for the preceding interval. All values are rates listed as events per second unless otherwise noted. When executing in a zone, and if the pools facility is active, mpstat(1M) will only provide information for those processors which are members of the processor set of the pool to which the zone is bound.

Column	Description
CPU or SET	Without the -a option, mpstat(1M) reports CPU statistics for a processor ID. With the -a option, mpstat(1M) reports SET statistics for a processor set ID.
minf	Minor faults.
mjf	Major faults.
xcal	Interprocessor cross-calls.
intr	Interrupts.
ithr	Interrupts as threads (not counting clock interrupt).

Column	Description
csw	Context switch.
icsw	Involuntary context switches.
migr	Thread migrations (to another processor).
smutx	Spins on mutexes (lock not acquired on first try).
srw	Spins on readers/writer locks (lock not acquired on first try).
syscall	System call.
usr	Percentage of user time.
sys	Percentage of system time.
wt	Time CPUs are idle pending I/O operations.
idl	Percentage of idle time.
sze	Number of processors in the requested processor set.
set	Processor set membership of each CPU.

Table 5: sar(1) General Format

Zone Process Monitoring

The ps(1) command prints information about active processes. Therefore, to find out the process count, use this:

```
ps -aef -o pid, zoneid,zone | grep <zonename> | wc -l
```

The proc tools are utilities that exercise features of /proc (see proc(4): http://docs.sun.com/app/docs/doc/816-5174/6mbb98uim?a=view). This takes a list of process-ids (pid). To discover all processes running within local zone, the utility ptree (1) prints the process trees containing the specified pids or users, with child processes indented from their respective parent processes. An argument of all digits is taken to be a process-id; otherwise, it is assumed to be a user login name. The default is all processes.

```
ptree -z <zonename>

ptree -z <zoneid>
```

This prints only the processes in the specified zone. Each zone ID can be specified as either a zone name or a numerical zone ID. This option is only useful when executed in the global zone.

The device and privilege restrictions have a number of effects on the utilities that can run in a non-global zone. The eeprom(1M), prtdiag(1M), and prtconf(1M) utilities do not work in a zone since they rely on devices that are not normally available.

For more information, see:

- `eeprom(1M)`: http://docs.sun.com/app/docs/doc/816-5166/6mbb1kq17?a=view
- `prtdiag(1M)`: http://docs.sun.com/app/docs/doc/816-5166/6mbb1kqcc?a=view
- `prtconf(1M)`: http://docs.sun.com/app/docs/doc/816-5166/6mbb1kqcb?a=view

Hence, the zone process count can be monitored as:

```
zlogin <zonename> ptree | wc -l
```

To examine the active processes on the system, use:

```
pgrep -flvx -z <zonename>
```

To discover the process privilege sets and attributes, use:

```
zlogin <zonename> ppriv -l zone
```

To display the scheduling parameters of a specific process, use:

```
priocntl -d -i zoneid <zoneid>
```

To find out zone-specific information, use:

```
ps -aef -z <zonename>
```

```
ps -aef -Z
```

Virtual Memory Statistics

The virtual memory system divides physical memory into segments known as pages. Pages are the fundamental unit of physical memory in the Solaris memory management subsystem. To read data from a file into memory, the virtual memory system reads in one page at a time, or pages in a file. To reduce resource consumption, the daemon can page out, or relocate, infrequently used pages to a swap device, which is an area outside of physical memory.

`vmstat(1M)` reports virtual memory statistics regarding kernel thread, virtual memory, disk, trap, and CPU activity. On MP (multiprocessor) systems, `vmstat(1M)` averages the number of CPUs into the output. For per-processor statistics, see `mpstat(1M)`. `vmstat(1M)` only supports statistics for certain devices. For more general I/O, use `iostat(1M)` instead.

Here is sample usage of a virtual memory snapshot for a specific zone:

```
zlogin <zonename> vmstat
```

Name	Description
kthr	r: the number of kernel threads in run queue.
	b: the number of blocked kernel threads that are waiting for resources I/O, paging, and so forth.
	w: the number of swapped-out lightweight processes (LWPs) that are waiting for processing resources to finish.
memory	Swap: available swap space (Kbytes).
	Free: size of the free list (Kbytes).
page	re: page reclaims.
	mf: minor faults.
	pi: Kbytes paged in.
	po: Kbytes paged out.
	fr: Kbytes freed.
	de: anticipated short-term memory shortfall (Kbytes).
	sr: pages scanned by clock algorithm.
	When executed in a zone and if the pools facility is active, all of the above only report activity on the processors in the processor set of the zone's pool.
disk	Report the number of disk operations per second.
faults	in: interrupts.
	sy: system calls.
	cs: CPU context switches.
	When executed in a zone and if the pools facility is active, all of the above only report activity on the processors in the processor set of the zone's pool.
cpu	Give a breakdown of percentage usage of CPU time. On MP systems, this is an average across all processors. When executed in a zone and if the pools facility is active, all of the above only report activity on the processors in the processor set of the zone's pool.

Table 6: vmstat(1M) General Format

A sample run looks like this:

```
zlogin webzone vmstat -S 5

kthr memory page disk faults cpu
```

```
r b w swap free si so pi po fr de sr s1 sd sd sd in sy cs us sy id

0 0 0 65024304 63837296 0 0 0 1 1 0 0 0 0 0 0 406 145 231 0 0 100

0 0 0 64490360 63267344 0 0 0 0 0 0 0 0 0 0 0 390 122 255 0 0 100
```

To find out page fault count, use the following.

`zlogin <zonename> vmstat 60` at the `mf` column shows the page fault count.

To display the total number of various system events since the system last booted, use:

```
zlogin <zonename> vmstat -s

0 swap ins

0 swap outs

0 pages swapped in

0 pages swapped out

14637968 total address trans. faults taken

54838 page ins

112523 page outs

77667 pages paged in

519846 pages paged out

2055772 total reclaims

1849074 reclaims from free list

0 micro (hat) faults

14637968 minor (as) faults

51558 major faults

2276335 copy-on-write faults

6057184 zero fill page faults

63484 pages examined by the clock daemon
```

```
0 revolutions of the clock hand

313118 pages freed by the clock daemon

54483 forks

13172 vforks

31340 execs

1108831482 cpu context switches

1943688216 device interrupts

16731856 traps

693614981 system calls

153906436 total name lookups (cache hits 99%)

506614 user cpu

12924434 system cpu

7650826588 idle cpu

0 wait cpu
```

Swap Statistics

The `swap(1M)` utility provides a method of adding, deleting, and monitoring the system swap areas used by the memory manager. `vmstat(1M)` also produces swap statistics.

```
zlogin <zonename> swap -l
```

List the status of all the swap areas. The output column looks like the following.

Name	Description
`path`	The path name for the swap area.
`dev`	The major/minor device number in decimal if it is a block special device; zeroes otherwise.
`swaplo`	The swaplow value for the area in 512-byte blocks.
`blocks`	The swaplen value for the area in 512-byte blocks.
`free`	The number of 512-byte blocks in this area that are not currently allocated.

Table 7: `swap(1M)` General Format

Sample output is as follows:

```
zlogin webzone swap -l

swapfile      dev    swaplo       blocks       free

/dev/swap     0,0    16           16403696     16403696
```

I/O Statistics

The `iostat(1M)` utility iteratively reports terminal, disk, and tape I/O activity, as well as CPU utilization. The first line of output is for all time since boot; each subsequent line is for the prior interval only. To compute this information, the kernel maintains a number of counters. For each disk, the kernel counts reads, writes, bytes read, and bytes written. The kernel also takes hi-res time stamps at queue entry and exit points, which allows it to keep track of the residence time and cumulative residence-length product for each queue. Using these values, `iostat(1M)` produces highly accurate measures of throughput, utilization, queue lengths, transaction rates, and service time. For terminals collectively, the kernel simply counts the number of input and output characters. During execution of the kernel status command, the state of the system can change. If relevant, a state change message is included in the `iostat(1M)` output.

Using `iostat(1M)` to Generate Partition and Device Statistics

The `iostat(1M)` utility can be used to report local zone-specific I/O statistics.

```
zlogin <zonename> iostat -xnp
```

Name	Description
`device`	Name of the disk.
`r/s`	Reads per second.
`w/s`	Writes per second.
`kr/s`	Kilobytes read per second. The average I/O size during the interval can be computed from `kr/s` divided by `r/s`.
`kw/s`	Kilobytes written per second. The average I/O size during the interval can be computed from `kw/s` divided by `r/s`.
`wait`	Average number of transactions waiting for service (queue length). This is the number of I/O operations held in the device driver queue waiting for acceptance by the device.
`actv`	Average number of transactions actively being serviced (removed from the queue but not yet completed). This is the number of I/O operations accepted, but not yet serviced, by the device.
`svc_t`	Average response time of transactions in milliseconds.
`%w`	Percentage of time there are transactions waiting for service (queue non-empty).

Name	Description
%b	Percentage of time the disk is busy (transactions in progress).
wsvc_t	Average service time in wait queue, in milliseconds.
asvc_t	Average service time of active transactions in milliseconds.
wt	The amount of time that CPUs are idle, pending I/O operations.

Table 8: iostat(1M) General Format

Please note: When executed in a zone and if the pools facility is active, iostat(1M) will only provide information for those processors in the processor set of the pool to which the zone is bound.

An example of output looks like this:

```
zlogin webzone iostat -xnp

       extended device statistics

r/s w/s kr/s kw/s wait actv wsvc_t asvc_t %w %b device
0.0 0.0 0.0  0.0  0.0  0.0  0.0    0.7    0  0  sd1
0.0 0.0 0.0  0.0  0.0  0.0  0.0    0.0    0  0  ssd0
0.0 0.0 0.0  0.0  0.0  0.0  0.0    0.0    0  0  ssd0,a
0.0 0.0 0.0  0.0  0.0  0.0  0.0    0.0    0  0  ssd0,b
0.0 0.0 0.0  0.0  0.0  0.0  0.0    0.0    0  0  ssd0,c
0.0 0.0 0.0  0.0  0.0  0.0  0.0    0.0    0  0  ssd0,g
0.0 0.0 0.0  0.0  0.0  0.0  0.0    0.0    0  0  ssd1
```

svc_t is the average response time of transactions, in milliseconds. The svc_t output reports the overall response time, rather than the service time, of a device. The overall time includes the time that transactions are in the queue and the time that transactions are being serviced. The time spent in the queue is shown with the -x option in the wsvc_t output column. The time spent servicing transactions is the true service time. Service time is also shown with the -x option and appears in the asvc_t output column of the same report.

%w is the percentage of time there are transactions waiting for service (queue non-empty).

%b is the percentage of time the disk is busy (transactions in progress).

wsvc_t is the average service time in wait queue, in milliseconds.

asvc_t is the average service time of active transactions, in milliseconds.

In addition, use iostat(1M) to generate user and system operation statistics:

```
zlogin <zonename> iostat -xcnCXTdz 5
```

The following command displays two reports of extended device statistics, aggregated by controller ID for user (us) and system (sy) operations.

To use iostat(1M) to generate TTY statistics, type the following:

```
zlogin <zonename> iostat -xtc 5 2
```

High wt times indicate problems in the disk subsystem, not problems with CPUs or other processing elements. Excessive wt times must be addressed by improving the performance, especially the service times, of the busiest disk devices.

To find out disk service time and busy time, use this command:

```
zlogin <zonename> iostat -xnp
```

Disk Usage

The df(1M) utility displays the amount of disk space occupied by mounted or unmounted file systems, the amount of used and available space, and how much of the file system's total capacity has been used. The file system is specified by device, or by referring to a file or directory on the specified file system. This displays the mount points of the file system in all visible zones. By default, df(1M) only displays mounts located within the current zone.

To display all visible zones, use:

```
df -Zkh
```

To display a specific local zone, use:

```
zlogin <zonename> df -kh
```

Sample output looks like this:

```
Filesystem   size used avail capacity  Mounted on

/            127G 6.5G 119G  6%        /

/dev         127G 6.5G 119G  6%        /dev

/lib         127G 6.5G 119G  6%        /lib

/platform    127G 6.5G 119G  6%        /platform

/sbin        127G 6.5G 119G  6%        /sbin

/us          127G 6.5G 119G  6%        /usr
```

```
proc          0K   0K   0K    0%        /proc

ctfs          0K   0K   0K    0%        /system/contract

swap          62G  272K 62G   1%        /etc/svc/volatile

mnttab        0K   0K   0K    0%        /etc/mnttab

fd            0K   0K   0K    0%        /dev/fd

swap          62G  0K   62G   0%        /tmp

swap          62G  32K  62G   1%        /var/run
```

Inode Usage on the File System

To display inode usage on all UFS file systems, use:

```
zlogin <zonename> df -F ufs -o i
```

Sample output looks like this:

```
Filesystem iused    ifree        %iused Mounted on

/          220891 15763621       1%     /
```

Network Interface

The command `ifconfig(1M)` is used to assign an address to a network interface and to configure network interface parameters. The `ifconfig(1M)` command must be used at boot time to define the network address of each interface present on a machine; it may also be used at a later time to redefine an interface's address or other operating parameters.

If no option is specified, `ifconfig(1M)` displays the current configuration for a network interface. If an address family is specified, `ifconfig(1M)` reports only the details specific to that address family. The `ifconfig(1M)` command does support flag zone `zonename`, which places the logical interface in the local zone. The named zone must be active in the kernel in the ready or running state. The interface is unplumbed when the zone is halted or rebooted. In addition, use `ifconfig(1M)` with the `-Z` flag to apply commands to all interfaces in the user's zones. There is one virtual interface alias per zone. A source address from the virtual interface alias in the same zone is selected. The virtual interface aliases were created using `zonecfg(1M)` (see http://docs.sun.com/app/docs/doc/816-5166/6mbb1kqlb?a=view).

```
zlogin <zonename> ifconfig -a
```

A sample interface configuration looks like the following:

```
zlogin webzone ifconfig -a

lo0:1: flags=2001000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv4,VIRTUAL> mtu 8232 ind

ex 1

        inet 127.0.0.1 netmask ff000000

eri0:1: flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2

        inet 192.168.1.1 netmask ffffff00 broadcast 192.168.1.255
```

Network Statistics

The `netstat(1M)` command displays the contents of certain network-related data structures in various formats, depending on the options you select.

```
netstat -s
```

The command shows the RAWIP, UDP, TCP, IPv4, IPv6, ICMPv4, ICMPv6, IGMP, and STCP statistics. Network resource usages are measured as shared resources at global level.

To check if IP forwarding is disabled, use:

```
netstat -s | grep ipForwProhibits
```

To find out an IP forwarding occurrence, use:

```
netstat -s | grep ipForwarding
```

To check TCP active opens, use:

```
netstat -s | grep tcpActiveOpens
```

To check TCP passive opens, use:

```
netstat -s | grep tcpPassiveOpens
```

To check retransmit occurrences, use:

```
netstat -s | grep tcpRetransBytes
```

To find the outbound interface errors, use:

```
netstat -s | grep tcpInErrs
```

IPC Status

The `ipcs(1)` utility reports data about active interprocess communication facilities. The information that is displayed is controlled by the options supplied. Without options, information is printed in short format for message queues, shared memory, and semaphores that are currently active in the system.

```
ipcs -A -z <zonename>
```

```
ipcs -A -Z
```

System Contract

The `ctstat(1)` utility allows a user to observe the contracts active on a system:

```
ctstat -a -z
```

Column Name	Description
CTID	The contract ID of the contract.
ZONEID	The zone ID of the contract's creator.
Type	Contract type.
State	The state of the contract.
Holder	If the contract is in the `owned` state, the PID of the process that owns the contract. If the contract is in the `inherited` state, the ID of the regent process contract.
Event	The number of unacknowledged critical events pending.
QTime	The time until quantum ends, or `-`, if no negotiation is in progress.
NTime	The time until negotiation ends, or `-`, if no negotiation is in progress.

Table 9: `ctstat(1)` General Format

Resource Management

Resource management is designed to optimize resource utilization on the Solaris platform via constraint mechanisms with resource bound specification, scheduling mechanisms for CPU resources, and partition mechanisms with zone and resource pool bound resource usage. A workload is an aggregation of all processes of an application or group of applications. Solaris resource management solutions offer performance tools to view the

current resource consumption of workloads that are running on your system. An administrator can then evaluate whether you must restrict access to a given resource or isolate particular workloads from other workloads.

To view defined projects, use:

```
projects -l
```

The `projects(1)` command prints on standard output the projects to which the invoking user or an optionally specified user belongs. Each user belongs to some set of projects.

To find out all the controllable resources in the zone, type:

```
prctl -i zone <zoneid>
```

Project Process Monitoring

To find out process-related information pertaining to the project or zone, use:

```
ps -aef -o user,pid,uid,projid,project,zoneid,zone
```

To find out the process count per project, use:

```
pgrep -J <projectID> | wc -l
```

To find out the process count per task, use:

```
pgrep -J <taskID> | wc -l
```

To find out the project count per zone, use:

```
ps -aef -o projid,project,zoneid,zone | grep <zonename> | grep <projectname> |
wc -l
```

To find out the task count per zone:

```
ps -aef -o taskid,zoneid,zone | grep <zonename> | grep <taskid> | wc -l
```

The `prstat(1M)` command described in the `prstat(1M)` man page monitors CPU usage by active projects (see: http://docs.sun.com/app/docs/doc/816-5166). The extended accounting data for tasks can be obtained for per-project statistics based on the amount of CPU resources that are consumed over longer periods.

To find out general resource usage per project, type:

```
prstat -J
```

Here is a sample of output:

```
PROJID NPROC SIZE RSS     MEMORY TIME   CPU   PROJECT
10     52     400M 271M    68%    0:11.45 0.4% webtier
0      35     113M 129M    32%    0:10.46 0.2% system
```

To find out general resource usage per task, use:

```
prstat -T
```

Here is a sample of output:

```
TASKID NPROC SIZE RSS     MEMORY TIME   CPU   PROJECT
222    30     229M 161M    44%    0:05:54 0.6% group.staff
223    1      26M  20M     5.3%   0:03:18 0.6% group.staff
12     1      61M  33M     8.9%   0:00:31 0.0% group.staff
1      33     85M  53M     14%    0:03:33 0.0% system
```

Activate the extended accounting facility to monitor and record resource consumption on a task or process basis. Use extended accounting data to assess current resource controls and to plan capacity requirements for future workloads. Aggregate usage on a system-wide basis can be tracked. To obtain complete usage statistics for related workloads that span more than one system, the project name can be shared across several machines. This process enables offline workload analysis to be correlated with online monitoring.

If it is required to determine whether a web server possesses sufficient CPUs for its typical workload, Solaris utility tools such as `sar(1)` and `prstat(1M)` can be used to collect data for idle CPU time and load average. In addition, extended accounting data needs to be examined to determine the number of simultaneous processes that are running for the web server process.

The `prctl(1M)` utility is designed to report the resource controls of running processes, tasks, and projects. The `prctl` utility allows the examination of the resource controls associated with an active process, task, or project on the system. It allows access to the basic and privileged limits on the specified entity.

To report all controls, use:

```
prctl -i zone <zonename>
```

To display the specific control constraint, use:

```
prctl -n zone.cpu-shares -i zone <zonename>
zone: 5: webzone
NAME              PRIVILEGE       VALUE    FLAG   ACTION  RECIPIENT
zone.cpu-shares   privileged          5       -   none  -
                  system          65.5K     max   none  -
```

Use the `prctl(1M)` command to place a privileged (superuser-owned) resource control on the tasks that contain specific user processes with particular thresholds.

```
prctl -n <taskattribute> -v <threadLimit> -t privileged -d all `pgrep
<processname>`
```

To enable a system log global action on the `task` attribute for resource control, use:

```
rctladm -e syslog task.max-lwps
```

The violation of the threshold will be logged as a warning in the system log.

See `/var/adm/messages` for the "exceeding" warning.

Resource Utilization With Cap Enforcement

The resource capping daemon `rcapd(1M)` enables you to regulate physical memory consumption by processes running in projects that have resource caps defined. The resource capping daemon `rcapd(1M)` manages physical memory by regulating the size of a project workload's resident set relative to the size of its working set. The resident set is the set of pages that are resident in physical memory. The working set is the set of pages that the workload actively uses during its processing cycle. The working set changes over time, depending on the process's mode of operation and the type of data being processed. Ideally, every workload has access to enough physical memory to enable its working set to remain resident. However, the working set can also include the use of secondary disk storage to hold the memory that does not fit in physical memory.

If you are using `rcapd(1M)` in a zones environment, you must add a `project` entry and configure the daemon in each zone where you want the daemon to run. `rcapd(1M)` will not act on processes in zones other than the one in which it is running.

To report resource cap enforcement daemon statistics, the `rcapstat(1)` command reports on the projects capped by `rcapd(1M)`. Each report contains statistics that pertain to the project and paging statistics. Paging refers to the act of relocating portions of memory, called pages, to or from physical memory. The `rcapd(1M)` pages out the most infrequently used pages. The paging statistics in the first report issued show the activity since the daemon was started. Subsequent reports reflect the activity since the last report was issued. Reports are issued every interval seconds up to the quantity specified by the count, or forever if the count is not specified.

To report cap and project information, use:

```
rcapstat 5 5
```

```
id      project     nproc  vm    rss   cap at  avgat pg   avgpg
112270 gproject1    24     123M   35M   50M 50M 0K   3312K 0K
78194  gproject2    1      2368K  1856K 10M 0K  0K   0K    0K

zlogin webzone rcapstat

id     project      nproc  vm rss cap   at avgat     pg avgpg

100    project1     9      64M 31M 32M  0K 0K        0K 0K

100    project1     9      64M 31M 32M  0K 0K        0K 0K

100    workproj1    9      64M 31M 32M  0K 0K        0K 0K
```

To monitor the RSS of a project, use:

```
rcapstat 5 5
id      project     nproc  vm   rss cap at    avgat  pg     avgpg
376565  gproject1   57     209M 46M 10M 440M   220M   5528K  2764K
376565  gproject2   57     209M 44M 10M 394M   131M   4912K  1637K
```

The project `gproject1` has an RSS in excess of its physical memory cap. The nonzero values in the `pg` column indicate that `rcapd(1M)` is consistently paging out memory as it attempts to meet the cap by lowering the physical memory utilization of the project's processes. However, `rcapd(1M)` is unsuccessful, as indicated by the varying `rss` values that do not show a corresponding decrease. This means that the application's resident memory is being actively used, forcing `rcapd(1M)` to affect the working set. Under this condition, the system continues to experience high page fault rates, and associated I/O, until the working set size (WSS) is reduced.

To report the working set size of a project, use:

```
rcapstat 5 5
id project nproc vm rss cap at avgat pg avgpg
376565 user1 56 207M 44M 10M 381M 191M 15M 7924K
376565 user1 56 207M 46M 10M 479M 160M 2696K 898K
```

By inhibiting cap enforcement, either by raising the cap of a project or by changing the minimum physical memory utilization for cap enforcement, the resident set can become the working set. The `rss` column might stabilize to show the project WSS, as shown in the previous example. The WSS is the minimum cap value that allows the project's processes to operate without perpetually incurring page faults.

To report memory utilization and the cap enforcement line, use:

```
rcapstat -g

id     project nproc vm rss cap at avgat pg avgpg
376565 rcap    0     0K 0K 10G  0K 0K    0K 0K
physical memory utilization: 55% cap enforcement threshold: 0%
```

```
id    project nproc vm rss cap at avgat pg avgpg
376565 rcap   0      0K 0K 10G 0K  0K    0K 0K
physical memory utilization: 55% cap enforcement threshold: 0%
```

DRP Resource Management

Solaris resource pools are used for partitioning machine resources. Dynamic resource pools (DRPs) dynamically adjust each resource pool's resource allocation to meet established system goals. On a system that has zones enabled, a non-global zone can be associated with one resource pool, although the pool need not be exclusively assigned to a particular zone. Moreover, you cannot bind individual processes in non-global zones to a different pool by using the `poolbind(1M)` command from the global zone. To associate a non-global zone with a pool, using the `zonecfg` subcommand, set `pool` to a specific resource pool. Then the reconfigured zone needs to be rebooted.

To list all instances of resource pool configuration, use:

```
pooladm
```

The `poolstat(1M)` utility run in a non-global zone displays only information about the pool associated with the zone. The `pooladm(1M)` command run without arguments in a non-global zone displays only information about the pool associated with the zone.

The `poolstat(1M)` utility includes options that can be used to examine specific pools and report resource set-specific statistics.

If zones are implemented on your system, and you use `poolstat(1M)` in a non-global zone, information about the resources associated with the zone's pool is displayed.

Name	Description
`id`	Pool ID.
`pool`	Pool name.
`rid`	Resource set ID.
`rset`	Resource set name.
`type`	Resource set type.
`min`	Minimum resource set size.
`max`	Maximum resource set size.
`size`	Current resource set size.
`used`	Measure of how much of the resource set is currently used.
`load`	Absolute representation of the load that is put on the resource set.

Table 10: `poolstat(1M)` General Format

Producing Multiple Reports at Specific Intervals

The `poolstat` utility shows all active pools on the system. It reports statistics based on the selected specific intervals with the number of counts.

```
poolstat 5 3

                          pset
id pool             size used load
46 pool_sales          2 1.2 8.3
0 pool_default         2 0.4 5.2
                          pset
id pool             size used load
46 pool_sales          2 1.4 8.4
0 pool_default         2 1.9 2.0
                          pset
id pool             size used load
46 pool_sales          2 1.1 8.0
0 pool_default         2 0.3 5.0
```

Reporting Resource Set Statistics

The following example uses the `poolstat(1M)` command with the `-r` option to report statistics for the processor-set resource set. Note that the resource set `pset_default` is attached to more than one pool, so this processor set is listed once for each pool membership.

```
poolstat -r pset
       id pool        type rid rset         min max size used load
       0 pool_default pset -1  pset_default 1  65K  2    1.2  8.3
       6 pool_sales   pset  1  pset_sales   1  65K  2    1.2  8.3
       2 pool_other   pset -1  pset_default 1  10K  2    0.4  5.2
```

Extended Accounting Facility

Without arguments, `acctadm(1M)` displays the current status of the extended accounting facility.

The following command displays resource groups for task accounting:

```
acctadm -r
```

Conclusion

All these utilities in the Solaris OS give administrators, system monitoring solution providers, and data center operators flexible instrumentation at system run time to identify performance bottleneck and service-level issues. The integration of these utilities with traditional system management solutions can provide realtime reports on data center operations.