



# Performance Monitoring (focused on Java SE)



# Topics

- What information to monitor
- What tools to use
- What level of the software stack to monitor
  - > Operating system level
  - > JVM level
  - > Application level
- What to monitor is covered per component at a given level
  - > Example. At OS level, monitor CPU usage

# Tools For Monitoring

- Definition: An act of non-intrusively collecting or observing performance data from an operating or running application.
- What to monitor and where?
  - > Operating system: cpu utilization (kernel & user), network i/o, disk i/o, memory, processes and kernel (locks).
  - > JVM: garbage collection frequency and duration, heap usage, threads, lock contention, cpu usage
  - > Application: throughput, responsiveness

# CPU Monitoring Tools at the OS-Level

# Tools For Monitoring : OS Level

- cpu
  - > vmstat (Solaris & Linux)
  - > mpstat (Solaris)
  - > prstat (Solaris)
  - > top (Linux, prefer prstat on Solaris)
  - > Task Manager (Windows)
  - > Performance Monitor (Windows)
  - > xosview (Linux)
  - > cpubar (Solaris – Performance Tools CD)
  - > iobar (Solaris – Performance Tools CD)
  - > dtrace (Solaris)

# Tools For Monitoring : vmstat

```

huntch@ditka: ~
File Edit View Terminal Tabs Help
kthr      memory          page        disk        faults        cpu
r  b  w  swap free re  mf pi po fr de sr m0 m1 m2 m3  in  sy  cs us sy id
0  0  0 11532280 3322672 4 18 6 0 0 0 0 1 1 1 0 1599 491 147 1 1 98
0  0  0 11406040 3186248 0 7 0 0 0 0 0 0 0 0 0 1567 348 138 0 1 99
0  0  0 11406832 3187048 0 1 0 0 0 0 0 19 19 19 0 1737 332 153 0 1 99
0  0  0 11408128 3188344 0 1 0 2 2 0 0 0 0 0 0 1573 336 134 0 1 99
0  0  0 11407912 3188128 0 1 0 0 0 0 0 0 0 0 0 1567 331 139 0 1 99
0  0  0 11407464 3187696 2 21 0 0 0 0 0 17 17 17 0 1717 331 157 0 1 99
0  0  0 11407472 3187704 0 1 0 0 0 0 0 0 0 0 0 1563 310 149 0 1 99
0  0  0 11407520 3187744 0 2 0 0 0 0 0 0 0 0 0 1565 317 144 0 1 99
0  0  0 11407504 3187720 0 1 0 0 0 0 0 0 0 0 0 1561 308 138 0 1 99
0  0  0 11407016 3187216 32 196 0 0 0 0 0 0 0 0 0 2764 4997 1381 1 2 97
0  0  0 11396280 3176232 43 351 0 0 0 0 0 0 0 0 0 5036 14128 3772 9 4 87
0  0  0 11399760 3179872 25 65 0 0 0 0 0 8 8 8 0 1740 2450 314 16 2 82
0  0  0 11407392 3187600 0 1 0 0 0 0 0 0 0 0 0 1579 332 150 17 1 82
0  0  0 11407352 3187568 0 3 0 0 0 0 0 0 0 0 0 1572 313 138 17 1 82
0  0  0 11407312 3187536 0 1 0 0 0 0 0 0 0 0 0 1573 324 143 12 1 86
0  0  0 11407280 3187504 0 1 0 2 2 0 0 0 0 0 0 1564 316 142 0 1 99
0  0  0 11407248 3187464 0 1 0 0 0 0 0 0 0 0 0 1562 319 147 0 1 99
0  0  0 11407216 3187440 0 1 0 0 0 0 0 3 3 3 0 1593 306 140 0 1 99
kthr      memory          page        disk        faults        cpu
r  b  w  swap free re  mf pi po fr de sr m0 m1 m2 m3  in  sy  cs us sy id
0  0  0 11407408 3187712 0 4 0 0 0 0 0 0 0 0 0 1583 557 179 0 1 98

```

# vmstat

- Virtual Memory Statistics
- reports virtual memory statistics of process, virtual memory, disk, trap, and CPU activity.
- cpu (breakdown of percentage usage of CPU time. On multiprocessors this is an average across all processors.)
  - > us - user time
  - > sy - system time
  - > id - idle time

# Tools For Monitoring : mpstat

huntch@ditka: ~

File	Edit	View	Terminal	Tab	Help										
4	0	0	0	101	100	10	0	0	0	0	1	0	1	0	99
5	131	0	550	105	100	1246	4	0	1	0	4877	42	10	0	48
8	1	0	0	101	100	53	0	1	0	0	355	0	1	0	98
9	23	0	0	102	100	363	0	0	0	0	1283	3	3	0	94
10	25	0	0	101	100	442	1	0	0	0	1426	3	4	0	93
11	0	0	0	2135	2134	34	0	0	0	0	15	0	8	0	92
12	0	0	0	101	100	35	0	1	1	0	4	0	1	0	99
13	0	0	0	101	100	4	0	0	0	0	1	0	1	0	99
14	0	0	62	102	101	8	0	0	0	0	1	0	1	0	99
15	0	0	0	101	100	9	0	0	0	0	0	0	2	0	98
CPU	minf	mjf	xcal	intr	ithr	csw	icsw	migr	smtx	srw	syscl	usr	sys	wt	idl
0	0	0	1107	400	300	15	0	0	0	0	7	0	2	0	98
1	0	0	0	101	100	13	0	0	0	0	0	0	1	0	99
4	0	0	0	101	100	11	0	0	0	0	1	0	1	0	99
5	46	0	470	103	100	117	2	0	0	0	1240	22	4	0	75
8	0	0	0	107	100	35	6	1	0	0	65	73	1	0	26
9	1	0	0	102	101	31	0	1	0	0	114	0	1	0	99
10	0	0	0	101	100	17	0	1	0	0	125	0	1	0	99
11	0	0	0	217	216	12	0	0	0	0	26	0	1	0	99
12	0	0	0	102	101	58	0	0	1	0	45	0	1	0	99
13	0	0	0	102	101	5	0	0	0	0	0	0	1	0	99
14	0	0	0	102	101	6	0	0	0	0	1	0	1	0	99
15	0	0	0	101	100	9	0	0	0	0	1	0	1	0	99

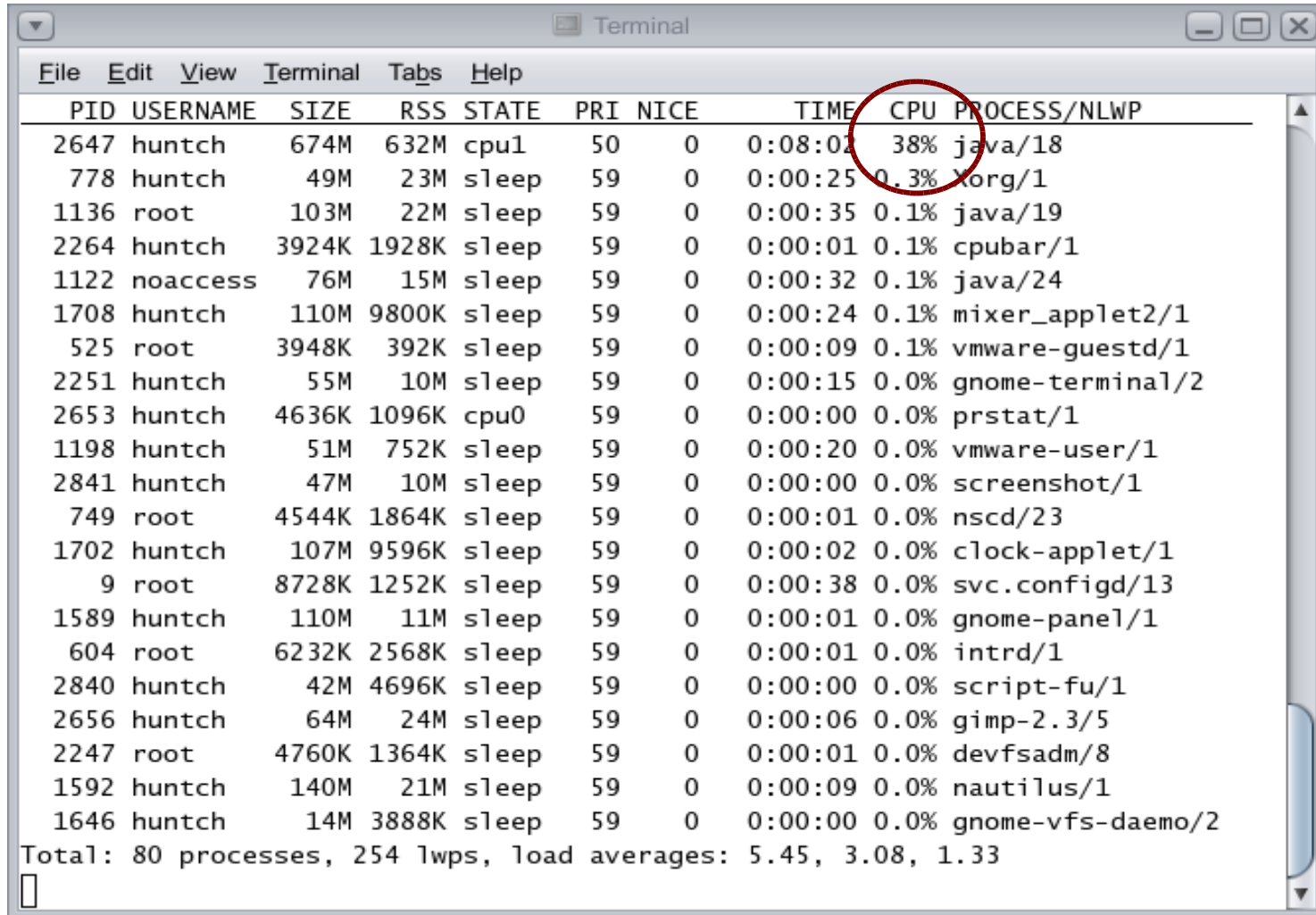
# mpstat

- Multi-processor status
- Reveals the individual CPU utilization on multi-processor
- 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 (events per second) unless otherwise noted.

# mpstat

- icsw involuntary context switches
- migr thread migrations (to another processor)
- usr percent user time
- sys percent system time

# Tools For Monitoring : prstat



PID	USERNAME	SIZE	RSS	STATE	PRI	NICE	TIME	CPU	PROCESS/NLWP
2647	huntch	674M	632M	cpu1	50	0	0:08:01	38%	java/18
778	huntch	49M	23M	sleep	59	0	0:00:25	0.3%	Xorg/1
1136	root	103M	22M	sleep	59	0	0:00:35	0.1%	java/19
2264	huntch	3924K	1928K	sleep	59	0	0:00:01	0.1%	cpubar/1
1122	noaccess	76M	15M	sleep	59	0	0:00:32	0.1%	java/24
1708	huntch	110M	9800K	sleep	59	0	0:00:24	0.1%	mixer_applet2/1
525	root	3948K	392K	sleep	59	0	0:00:09	0.1%	vmware-guestd/1
2251	huntch	55M	10M	sleep	59	0	0:00:15	0.0%	gnome-terminal/2
2653	huntch	4636K	1096K	cpu0	59	0	0:00:00	0.0%	prstat/1
1198	huntch	51M	752K	sleep	59	0	0:00:20	0.0%	vmware-user/1
2841	huntch	47M	10M	sleep	59	0	0:00:00	0.0%	screenshot/1
749	root	4544K	1864K	sleep	59	0	0:00:01	0.0%	nscd/23
1702	huntch	107M	9596K	sleep	59	0	0:00:02	0.0%	clock-applet/1
9	root	8728K	1252K	sleep	59	0	0:00:38	0.0%	svc.configd/13
1589	huntch	110M	11M	sleep	59	0	0:00:01	0.0%	gnome-panel/1
604	root	6232K	2568K	sleep	59	0	0:00:01	0.0%	intrd/1
2840	huntch	42M	4696K	sleep	59	0	0:00:00	0.0%	script-fu/1
2656	huntch	64M	24M	sleep	59	0	0:00:06	0.0%	gimp-2.3/5
2247	root	4760K	1364K	sleep	59	0	0:00:01	0.0%	devfsadm/8
1592	huntch	140M	21M	sleep	59	0	0:00:09	0.0%	nautilus/1
1646	huntch	14M	3888K	sleep	59	0	0:00:00	0.0%	gnome-vfs-daemo/2

Total: 80 processes, 254 lwps, load averages: 5.45, 3.08, 1.33

# Tools For Monitoring : prstat -Lm

```

Terminal
File Edit View Terminal Tabs Help
PID USERNAME USR SYS TRP TFL DFL LCK SLP LAT VCX ICX SCL SIG PROCESS/LWPID
2647 huntch 27 0.0 0.1 0.0 0.0 52 0.0 21 7 113 28 0 java/3
2647 huntch 20 0.1 0.1 0.0 0.0 48 0.0 32 48 218 103 0 java/46
2647 huntch 19 0.0 0.1 0.0 0.0 38 0.0 44 60 223 115 0 java/45
2647 huntch 18 0.0 0.0 0.0 0.0 48 0.0 34 70 114 128 0 java/40
2647 huntch 18 0.1 0.1 0.0 0.0 44 0.0 38 52 106 115 0 java/39
2647 huntch 18 0.1 0.1 0.0 0.0 38 0.0 44 69 152 153 0 java/43
2647 huntch 17 0.0 0.1 0.0 0.0 44 0.0 39 54 194 135 0 java/41
2647 huntch 17 0.0 0.1 0.0 0.0 43 0.0 40 60 166 94 0 java/42
2647 huntch 16 0.0 0.1 0.0 0.0 39 0.0 45 70 160 126 1 java/44
2647 huntch 0.1 0.1 0.0 0.0 0.0 0.0 98 2.0 144 0 87 0 java/10
2647 huntch 0.0 0.0 0.0 0.0 0.0 100 0.0 0.0 0 0 0 0 java/9
2647 huntch 0.0 0.0 0.0 0.0 0.0 100 0.0 0.0 0 0 0 0 java/8
2647 huntch 0.0 0.0 0.0 0.0 0.0 100 0.0 0.0 0 0 0 0 java/7
2647 huntch 0.0 0.0 0.0 0.0 0.0 100 0.0 0.0 0 0 0 0 java/6
2647 huntch 0.0 0.0 0.0 0.0 0.0 100 0.0 0.0 0 0 0 0 java/5
2647 huntch 0.0 0.0 0.0 0.0 0.0 100 0.0 0.0 0 0 0 0 java/4
2647 huntch 0.0 0.0 0.0 0.0 0.0 0.0 100 0.0 0 0 0 0 java/2
2647 huntch 0.0 0.0 0.0 0.0 0.0 0.0 100 0.0 0 0 0 0 java/1

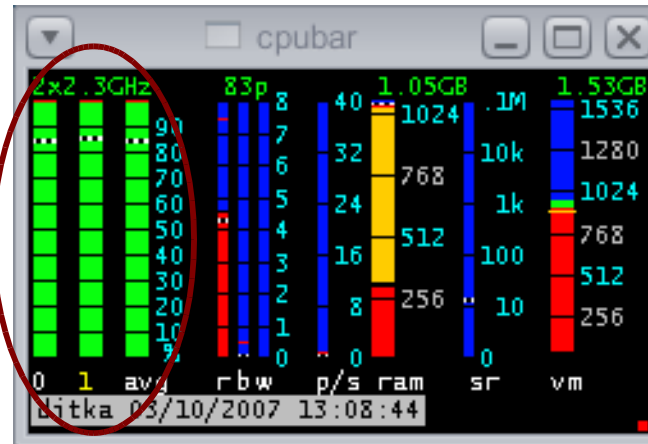
Total: 1 processes, 18 lwps, load averages: 5.61, 3.66, 1.68

```

# vmstat

- CPU The percentage of recent CPU time used by the process.
- VCX The number of voluntary context switches.
- ICX The number of involuntary context switches.

# Tools For Monitoring : cpubar



- Available on Solaris Performance Tools 3.0 CD, or download from:

<http://mediacast.sun.com/share/stefanschneider/PerformanceCD3.0.tar.gz>

# CPU monitoring : What to look for

- Using HotSpot's `jps` command you can find the process ids of all running Java processes on your machine.
- Using Solaris `prstat -Lm`, or `prstat -Lmp <pid>` to locate the LWP id(s) consuming the most cpu (usr or sys) and using HotSpot's `jstack` you can find the executing threads taking the cpu (usr and sys) time.
  - > LWPID is in the far right column of `prstat`.
  - > Look for `jstack`'s corresponding 'tid', reported in hex.

# CPU monitoring : What to look for

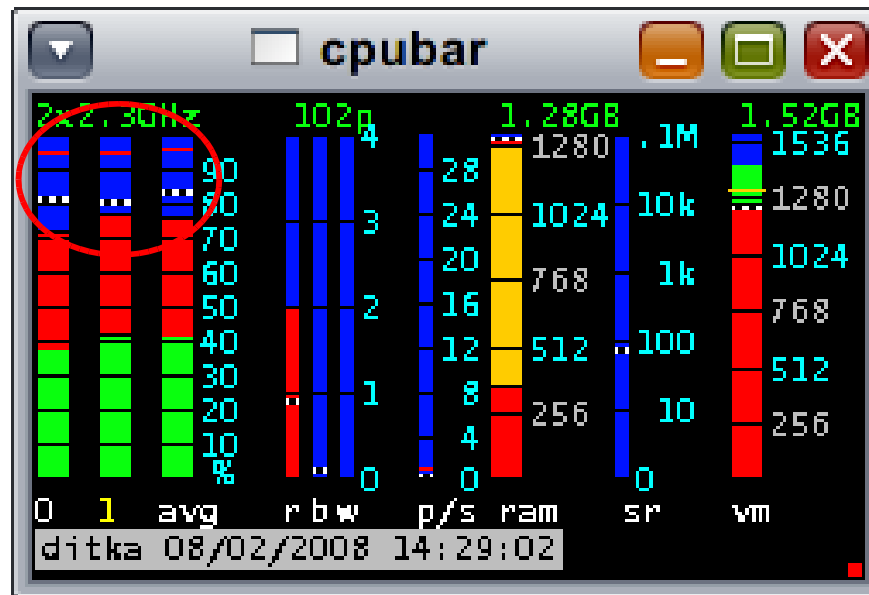
- Idle cpu
  - > On multi-threaded applications and multi-core systems, idle cpu can be an indicator of an application's inability to scale.
  - > Combination of high sys or kernel CPU utilization and idle CPU could indicate shared resource contention as the scalability blocker.
  - > Applicable to all operating systems, i.e. Windows, Linux and Solaris

# CPU monitoring : What to look for

```

Terminal
File Edit View Terminal Tabs Help
bash-3.2$ vmstat 5
kthr      memory          page        disk        faults        cpu
 r  b  w   swap  free  re  mf  pi  po  fr  de  sr  f0  s0  s1  s2   in   sy   cs  us  sy  id
0  0  0 659620 168400   3  19   6   2   3   0   4  -0   1  -0   1  385  886  432   3   2  95
0  0  0 129404 26456   2  87  10   0   0   0   0   0   5   0   0 2320 247894 4745  37  20  44
0  0  0 127920 25524  23 395  90   0   0   0   0   0  27   0   1 1844 241699 4280  45  21  35
0  0  0 126380 24204   1  92   0   0   0   0   0   0   0   0   0 2455 283529 4916  34  19  48
0  0  0 126380 24208   1  93   0   0   0   0   0   0   0   0   0 2469 288083 4933  33  19  49
0  0  0 126376 24204   1  79   0   0   0   0   0   0   0   0   0 2171 241732 4294  33  20  47
0  0  0 126344 24172   1  78   1   0   0   0   0   0  16   0   3 2202 242373 4782  36  20  44
0  0  0 126544 24372   1  85   0   0   0   0   0   0   0   0   0 2127 263456 4928  42  21  37
0  0  0 126504 24332   1  92   0   0   0   0   0   0   0   0   6 2498 284583 5041  33  19  48
0  0  0 126112 23940   1 144   2   0   0   0   0   0   0   0   2 2027 273269 5311  46  20  34
0  0  0 125252 22952   3  97  53   0   0   0   0   0  10   0  14 2091 231099 4423  35  20  44
  
```

# CPU monitoring : What to look for



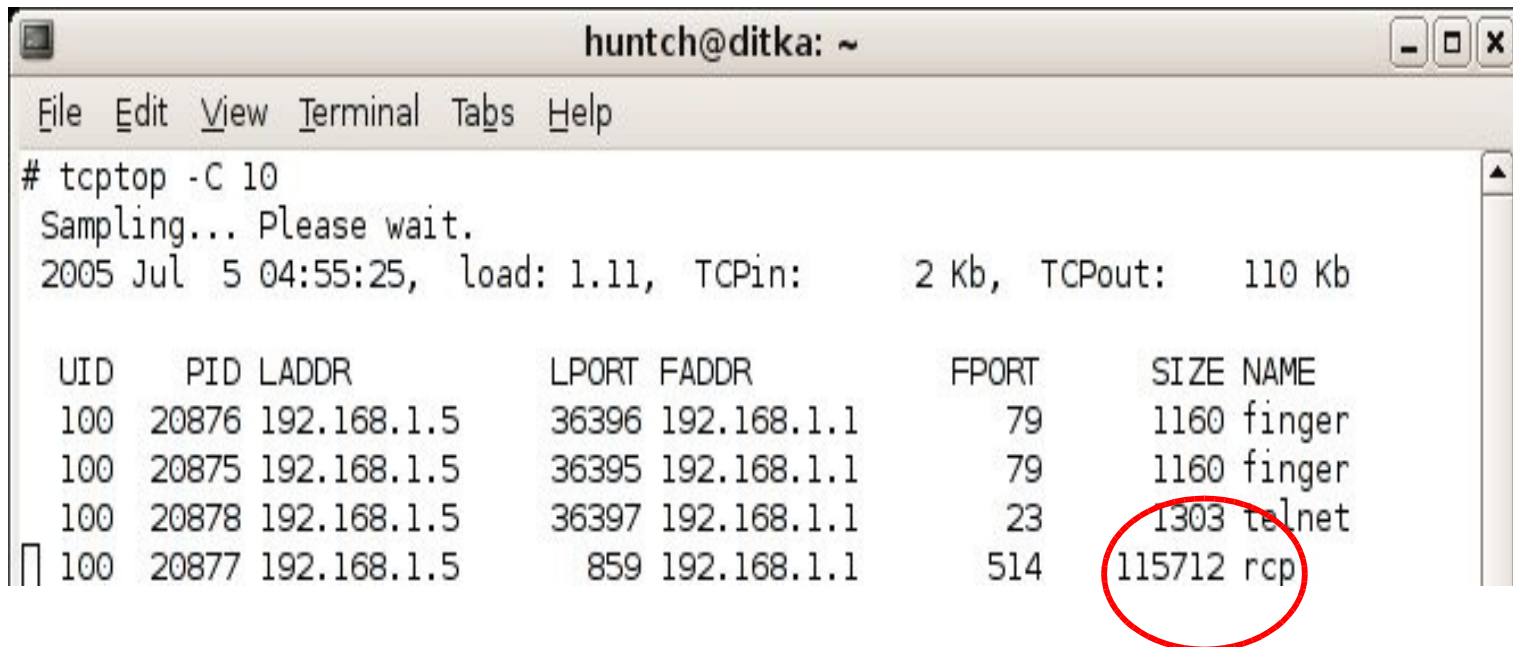
# Networking I/O Monitoring at OS Level

# Tools For Monitoring : OS Level

- network i/o
  - > netstat (Solaris & Linux)
  - > Performance Monitor (Windows)
  - > dtrace (Solaris)
  - > nicstat (Solaris – Performance Tools CD)
  - > tcptop (Dtrace Toolkit)
- Data of interest
  - > network utilization, established connections,

# Network i/o : What to look for

- tcptop can show per process tcp stats

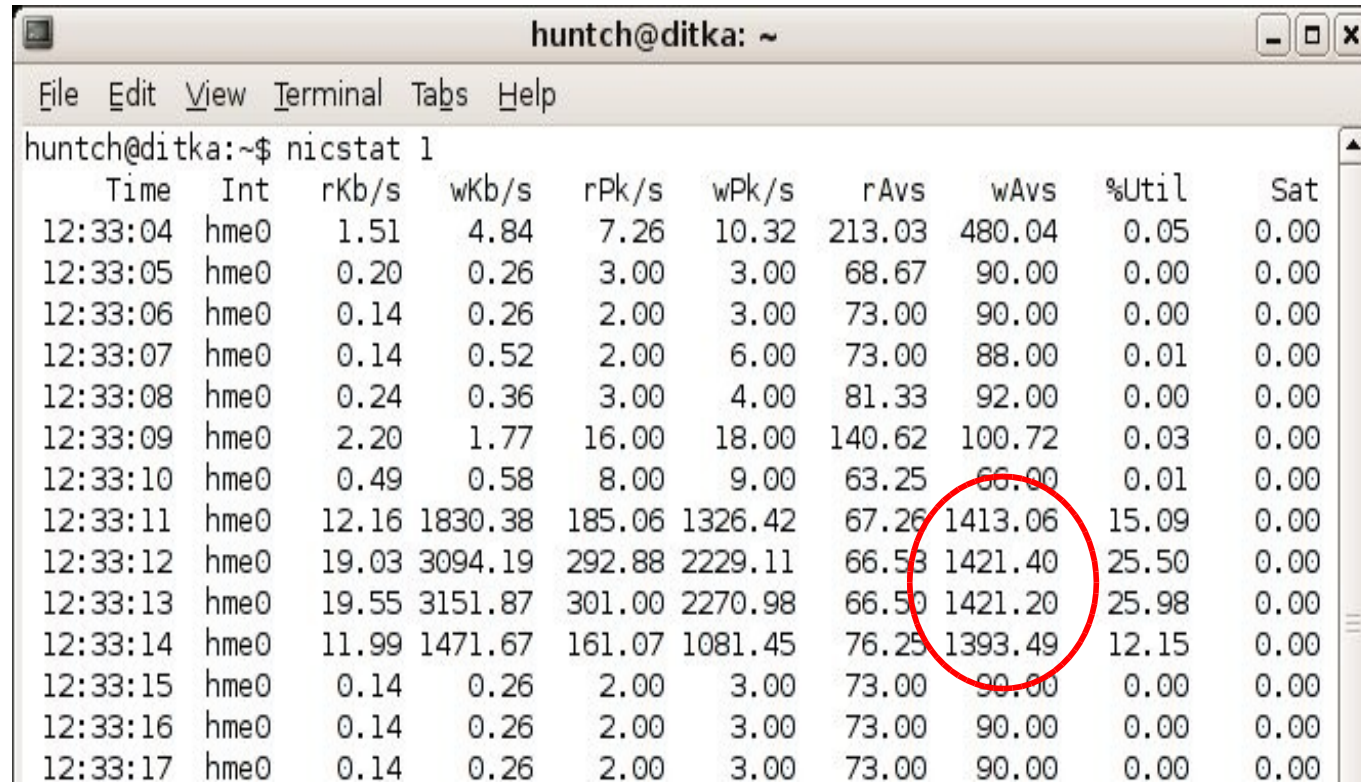


```
huntch@ditka: ~
File Edit View Terminal Tabs Help
# tcptop -C 10
Sampling... Please wait.
2005 Jul 5 04:55:25, load: 1.11, TCPin:      2 Kb, TCPout:    110 Kb

UID   PID  LADDR      LPORT  FADDR      FPORT   SIZE  NAME
100   20876 192.168.1.5 36396 192.168.1.1 79      1160  finger
100   20875 192.168.1.5 36395 192.168.1.1 79      1160  finger
100   20878 192.168.1.5 36397 192.168.1.1 23      1303  telnet
100   20877 192.168.1.5 859   192.168.1.1 514     115712 rcp
```

- Here we can see 'rcp' is generating about 115 kb of traffic

# Tools For Monitoring : nicstat



```

huntch@ditka:~$ nicstat 1
  Time  Int  rKb/s  wKb/s  rPk/s  wPk/s  rAvs  wAvs  %Util  Sat
12:33:04 hme0  1.51  4.84  7.26  10.32  213.03  480.04  0.05  0.00
12:33:05 hme0  0.20  0.26  3.00  3.00  68.67  90.00  0.00  0.00
12:33:06 hme0  0.14  0.26  2.00  3.00  73.00  90.00  0.00  0.00
12:33:07 hme0  0.14  0.52  2.00  6.00  73.00  88.00  0.01  0.00
12:33:08 hme0  0.24  0.36  3.00  4.00  81.33  92.00  0.00  0.00
12:33:09 hme0  2.20  1.77  16.00  18.00  140.62  100.72  0.03  0.00
12:33:10 hme0  0.49  0.58  8.00  9.00  63.25  66.00  0.01  0.00
12:33:11 hme0  12.16 1830.38 185.06 1326.42 67.26 1413.06 15.09 0.00
12:33:12 hme0  19.03 3094.19 292.88 2229.11 66.53 1421.40 25.50 0.00
12:33:13 hme0  19.55 3151.87 301.00 2270.98 66.50 1421.20 25.98 0.00
12:33:14 hme0  11.99 1471.67 161.07 1081.45 76.25 1393.49 12.15 0.00
12:33:15 hme0  0.14  0.26  2.00  3.00  73.00  90.00  0.00  0.00
12:33:16 hme0  0.14  0.26  2.00  3.00  73.00  90.00  0.00  0.00
12:33:17 hme0  0.14  0.26  2.00  3.00  73.00  90.00  0.00  0.00
  
```

- Notice wAvs, write average size, during four intervals is about 1420 bytes, the MTU size.

# Disk I/O Monitoring at OS Level

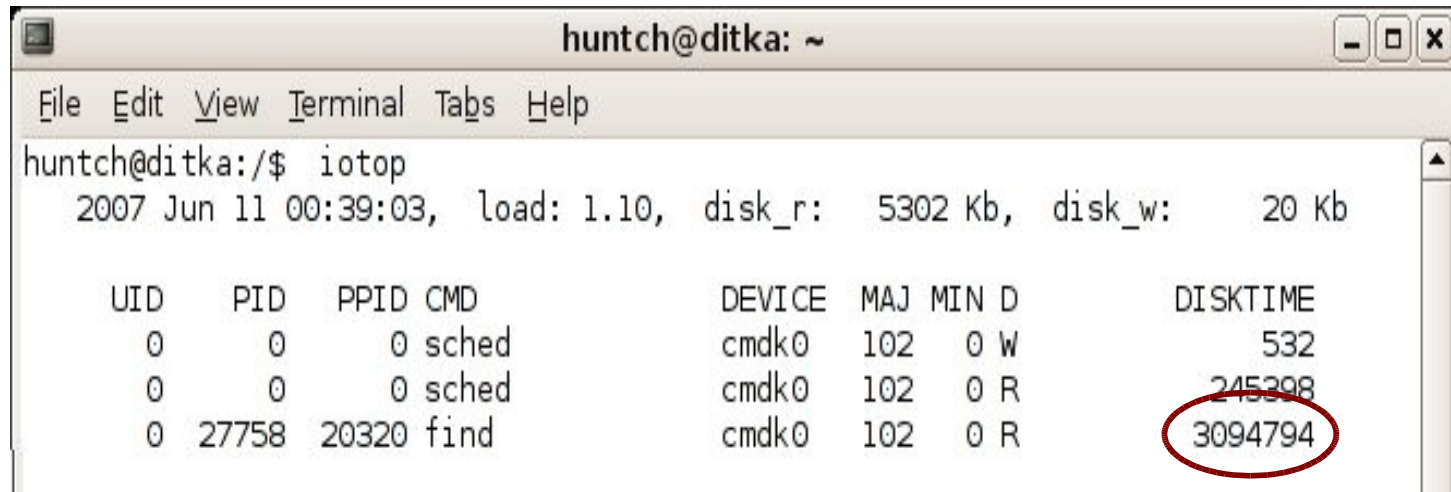
# Tools For Monitoring : OS Level

- disk i/o
  - > iostat (Solaris & Linux)
  - > Performance Monitor (Windows)
  - > dtrace (Solaris)
  - > iobar (Solaris – Performance Tools CD)
  - > iotop (Solaris – Performance Tools CD)
- Data of interest
  - > number of disk accesses, latency, average latencies

# Tools For Monitoring : iostat, iobar, iotop

- iostat reports per disk, text output
- iobar reports per disk, gui output
- iotop reports per process statistics, text output
- Data of interest
  - > number of disk accesses, latency, average latencies

# Tools For Monitoring : iotop example



```
huntch@ditka:/$ iotop
2007 Jun 11 00:39:03, load: 1.10, disk_r: 5302 Kb, disk_w: 20 Kb

  UID    PID    PPID  CMD          DEVICE MAJ MIN D    DISKTIME
   0      0      0  sched        cmdk0  102  0  W      532
   0      0      0  sched        cmdk0  102  0  R      245398
   0  27758  20320  find         cmdk0  102  0  R      3094794
```

- iotop reporting at a 5 second interval
- DISKTIME reported in microseconds
- CMD find is keeping disk cmdk0 busy almost 60% of time during the 5 second interval

# disk i/o : What to look for

- Disk cache
  - > Why not enable disk cache?
  - > What's the risk?
  - > On Solaris x86, disk cache may be disabled by default. Linux & Windows systems usually have it enabled.
  - > Disk cache being disabled depends on the Sun branded model and how recent the model.

# Memory Monitoring at OS Level

# Tools For Monitoring : OS Level

- memory
  - > vmstat (Solaris & Linux)
  - > prstat (Solaris)
  - > top (Linux – prefer prstat on Solaris)
  - > Performance Monitor (Windows)
  - > dtrace (Solaris)
  - > cpubar (Solaris – Performance Tools CD)
  - > meminfo (Solaris – Performance Tools CD)
- Data of interest
  - > paging, memory usage

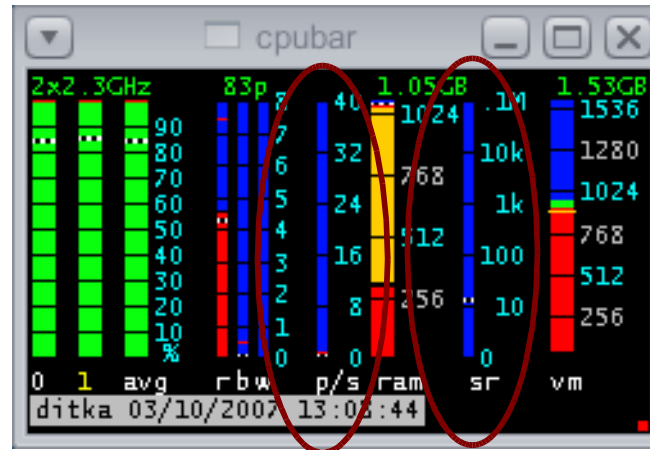
# Tools For Monitoring : vmstat

```

huntch@ditka: ~
File Edit View Terminal Tabs Help
kthr      memory          page            disk           faults         cpu
r  b  w  swap free re  mf pi po fr de sr m0 m1 m2 m3  in  sy  cs us sy id
0  0  0 11532280 3322672 4 18 6 0 0 0 0 1 1 1 0 1599 491 147 1 1 98
0  0  0 11406040 3186248 0 7 0 0 0 0 0 0 0 0 0 1567 348 138 0 1 99
0  0  0 11406832 3187048 0 1 0 0 0 0 0 19 19 19 0 1737 332 153 0 1 99
0  0  0 11408128 3188344 0 1 0 2 2 0 0 0 0 0 0 1573 336 134 0 1 99
0  0  0 11407912 3188128 0 1 0 0 0 0 0 0 0 0 0 1567 331 139 0 1 99
0  0  0 11407464 3187696 2 21 0 0 0 0 0 17 17 17 0 1717 331 157 0 1 99
0  0  0 11407472 3187704 0 1 0 0 0 0 0 0 0 0 0 1563 310 149 0 1 99
0  0  0 11407520 3187744 0 2 0 0 0 0 0 0 0 0 0 1565 317 144 0 1 99
0  0  0 11407504 3187720 0 1 0 0 0 0 0 0 0 0 0 1561 308 138 0 1 99
0  0  0 11407016 3187216 32 196 0 0 0 0 0 0 0 0 0 0 2764 4997 1381 1 2 97
0  0  0 11396280 3176232 43 351 0 0 0 0 0 0 0 0 0 5036 14128 3772 9 4 87
0  0  0 11399760 3179872 25 65 0 0 0 0 0 8 8 8 0 1740 2450 314 16 2 82
0  0  0 11407392 3187600 0 1 0 0 0 0 0 0 0 0 0 1579 332 150 17 1 82
0  0  0 11407352 3187568 0 3 0 0 0 0 0 0 0 0 0 1572 313 138 17 1 82
0  0  0 11407312 3187536 0 1 0 0 0 0 0 0 0 0 0 1573 324 143 12 1 86
0  0  0 11407280 3187504 0 1 0 2 2 0 0 0 0 0 0 1564 316 142 0 1 99
0  0  0 11407248 3187464 0 1 0 0 0 0 0 0 0 0 0 1562 319 147 0 1 99
0  0  0 11407216 3187440 0 1 0 0 0 0 0 3 3 3 0 1593 306 140 0 1 99
kthr      memory          page            disk           faults         cpu
r  b  w  swap free re  mf pi po fr de sr m0 m1 m2 m3  in  sy  cs us sy id
0  0  0 11407408 3187712 0 4 0 0 0 0 0 0 0 0 0 1583 557 179 0 1 98

```

# Tools For Monitoring : cpubar



- Data of interest
  - > p/s - pages per second, sr - scan rate
  - > Watch for high scan rate, or increasing trend. Low scan is ok if they are infrequent.

# memory utilization : What to look for

```

huntch@ditka: ~
File Edit View Terminal Tabs Help
huntch@ditka:~$ vmstat 5
kthr      memory          page        disk        faults       cpu
r  b  w    swap  free  re  mf  pi  po  fr  de  sr  f0  s0  s1  s2   in   sy   cs  us  sy  id
1  0  0 499792 154720  1 1697 0  0  0  0  0  0  0  12  811  612 1761 90  7  4
1  0  0 498856 44052  1 3214 0  0  0  0  0  0  0  12 1290 2185 3078 66 18 15
3  0  0 501188 17212  1 1400 2 2092 4911 0 37694 0 53 0 12 5262 3387 1485 52 27 21
1  0  0 500696 20344 26 2562 18 4265 7553 0 9220 0 66 0 12 1192 3007 2733 71 17 12
1  0  0 499976 20108  3 3146 24 3032 10009 0 10971 0 63 0 6 1346 1317 3358 78 15 7
1  0  0 743664 259080 61 1706 70 8882 10017 0 19866 0 178 0 52 1213 595 688 70 12 18
  
```

- Why is swapping bad for a Java application?
- Anyone volunteers want to explain?

# memory utilization : What to look for

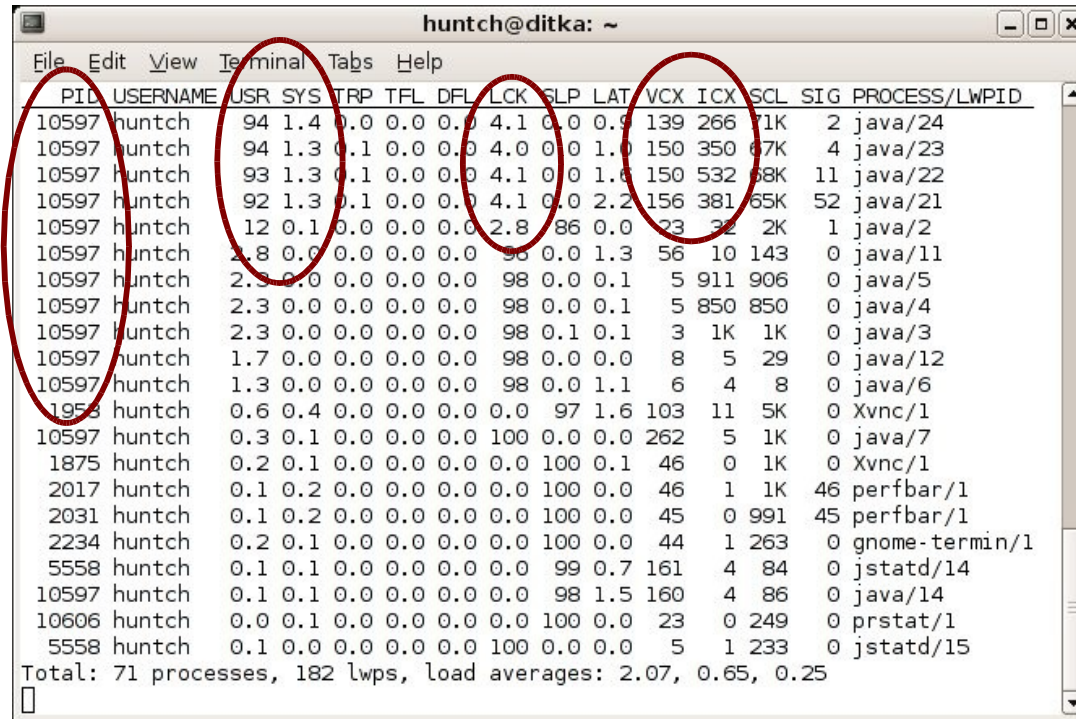
- How do you fix the swapping problem?
  - > Smaller Java heap sizes
  - > Add physical memory
  - > Reduce number of applications running on the machine
  - > Any one, or any combination of the above will help

# Processes Monitoring at OS Level

# Tools For Monitoring : OS Level

- processes
  - > ps (Solaris & Linux)
  - > vmstat (Solaris & Linux)
  - > mpstat (Solaris)
  - > prstat (Solaris)
  - > Performance Monitor (Windows)
  - > top (Linux – prefer prstat on Solaris)
  - > dtrace (Solaris)
- Data of interest
  - > footprint size, number of threads, thread state, cpu usage, runtime stack, context switches

# Tools For Monitoring : prstat -Lm



```

huntch@ditka: ~
File Edit View Terminal Tabs Help
PID USERNAME USR SYS TRP TFL DFL LCK SLP LAT VCX ICX SCL SIG PROCESS/LWPID
10597 huntch 94 1.4 0.0 0.0 0.0 4.1 0.0 0.9 139 266 71K 2 java/24
10597 huntch 94 1.3 0.1 0.0 0.0 4.0 0.0 1.0 150 350 67K 4 java/23
10597 huntch 93 1.3 0.1 0.0 0.0 4.1 0.0 1.6 150 532 68K 11 java/22
10597 huntch 92 1.3 0.1 0.0 0.0 4.1 0.0 2.2 156 381 65K 52 java/21
10597 huntch 12 0.1 0.0 0.0 0.0 2.8 0.0 0.0 86 0.0 23 32 2K 1 java/2
10597 huntch 2.8 0.0 0.0 0.0 0.0 98 0.0 1.3 56 10 143 0 java/11
10597 huntch 2.3 0.0 0.0 0.0 0.0 98 0.0 0.1 5 911 906 0 java/5
10597 huntch 2.3 0.0 0.0 0.0 0.0 98 0.0 0.1 5 850 850 0 java/4
10597 huntch 2.3 0.0 0.0 0.0 0.0 98 0.1 0.1 3 1K 1K 0 java/3
10597 huntch 1.7 0.0 0.0 0.0 0.0 98 0.0 0.0 8 5 29 0 java/12
10597 huntch 1.3 0.0 0.0 0.0 0.0 98 0.0 1.1 6 4 8 0 java/6
1958 huntch 0.6 0.4 0.0 0.0 0.0 0.0 97 1.6 103 11 5K 0 Xvnc/1
10597 huntch 0.3 0.1 0.0 0.0 0.0 100 0.0 0.0 262 5 1K 0 java/7
1875 huntch 0.2 0.1 0.0 0.0 0.0 0.0 100 0.1 46 0 1K 0 Xvnc/1
2017 huntch 0.1 0.2 0.0 0.0 0.0 0.0 100 0.0 46 1 1K 46 perfbar/1
2031 huntch 0.1 0.2 0.0 0.0 0.0 0.0 100 0.0 45 0 991 45 perfbar/1
2234 huntch 0.2 0.1 0.0 0.0 0.0 0.0 100 0.0 44 1 263 0 gnome-termin/1
5558 huntch 0.1 0.1 0.0 0.0 0.0 0.0 99 0.7 161 4 84 0 jstatd/14
10597 huntch 0.1 0.1 0.0 0.0 0.0 0.0 98 1.5 160 4 86 0 java/14
10606 huntch 0.0 0.1 0.0 0.0 0.0 0.0 100 0.0 23 0 249 0 prstat/1
5558 huntch 0.1 0.0 0.0 0.0 0.0 100 0.0 0.0 5 1 233 0 jstatd/15
Total: 71 processes, 182 lwps, load averages: 2.07, 0.65, 0.25

```

- Data of interest
  - > number of threads, cpu usage, locks, context switches

# Tools For Monitoring : mpstat

```

huntch@ditka: ~
File Edit View Terminal Tabs Help
huntch@ditka:~$ mpstat 5
CPU minf mjf xcal  intr ithr  csw icsw migr smtx  srw syscl  usr  sys  wt  idl
 0   21   1   6   319 133 261  18  23   3   0  551   4   3   0  92
 1   19   1   4   36  14  59  15  23   3   0  491   4   3   0  93
 2   17   1   4   19  12  61  13  23   3   0  355   4   3   0  90
 3   18   1   4   16  15  49  12  23   3   0  390   4   3   0  91
CPU minf mjf xcal  intr ithr  csw icsw migr smtx  srw syscl  usr  sys  wt  idl
 0   28   2   0  192  83  92  32  14   2   0  185  78  15   0   7
 1   49   1   0   37   1  80  28  16   2   0  139  80  16   0   4
 2   28   1   0   20   7  94  34  17   1   0  283  83  12   0   5
 3   39   1   2   52   1  99  36  16   3   0  219  74  19   0   7
CPU minf mjf xcal  intr ithr  csw icsw migr smtx  srw syscl  usr  sys  wt  idl
 0   34   0   2  171  75  78  32  12   1   0  173  90   9   0   2
 1   38   1   0   39   1  84  29  13   2   0  153  66  12   0  23
 2   28   8   0   21   9  97  31  20   2   0  167  67  13   0  20
 3   35   3   1   43   1  98  29  20   3   0  190  52  25   0  23

```

- Data of interest
  - > context switches, lock contention

# processes : What to look for

- Why footprint size, number of threads, thread state, lock contention and context switching are important to monitor?
- What does lock contention and/or context switching look like on Solaris?
- How can you find the lock or locks causing problems?
- How can you address the thread context switching problem?

# Kernel Monitoring at OS Level

# Tools For Monitoring : OS Level

- kernel
  - > vmstat (Linux & Solaris)
  - > mpstat (Solaris)
  - > lockstat & plockstat (Solaris)
  - > Performance Monitor (Windows)
  - > dtrace (Solaris)
  - > intrstat (Solaris)
- Data of interest
  - > kernel cpu utilization, locks, system calls, interrupts, migrations, run queue depth

# Tools For Monitoring : vmstat

```

Terminal
File Edit View Terminal Tabs Help
bash-3.2$ vmstat 5
kthr      memory          page        disk        faults        cpu
 r  b  w    swap free  re  mf pi po fr de sr f0 s0 s1 s2   in   sy   cs us sy id
0  0  0 659620 168400   3  19  6  2  3  0  4 -0  1 -0  1  385  886  432  3  2 95
0  0  0 129404 26456   2  87 10  0  0  0  0  0  5  0  0 2320 247894 4745  87 20 44
0  0  0 127920 25524  23 395 90  0  0  0  0  0 27  0  1 1844 241699 4280  45 21 35
0  0  0 126380 24204   1  92  0  0  0  0  0  0  0  0  0 2455 283529 4916  34 19 48
0  0  0 126380 24208   1  93  0  0  0  0  0  0  0  0  0 2469 288083 4933  33 19 49
0  0  0 126376 24204   1  79  0  0  0  0  0  0  0  0  0 2171 241732 4294  33 20 47
0  0  0 126344 24172   1  78  1  0  0  0  0  0 16  0  3 2202 242373 4782  36 20 44
0  0  0 126544 24372   1  85  0  0  0  0  0  0  0  0  0 2127 263456 4928  42 21 37
0  0  0 126504 24332   1  92  0  0  0  0  0  0  0  0  6 2498 284583 5041  33 19 48
0  0  0 126112 23940   1 144  2  0  0  0  0  0  0  0  2 2027 273269 5311  46 20 34
0  0  0 125252 22952   3  97 53  0  0  0  0  0 10  0 14 2091 231099 4423  35 20 44

```

- Data of interest
  - > kernel cpu utilization, run queue depth

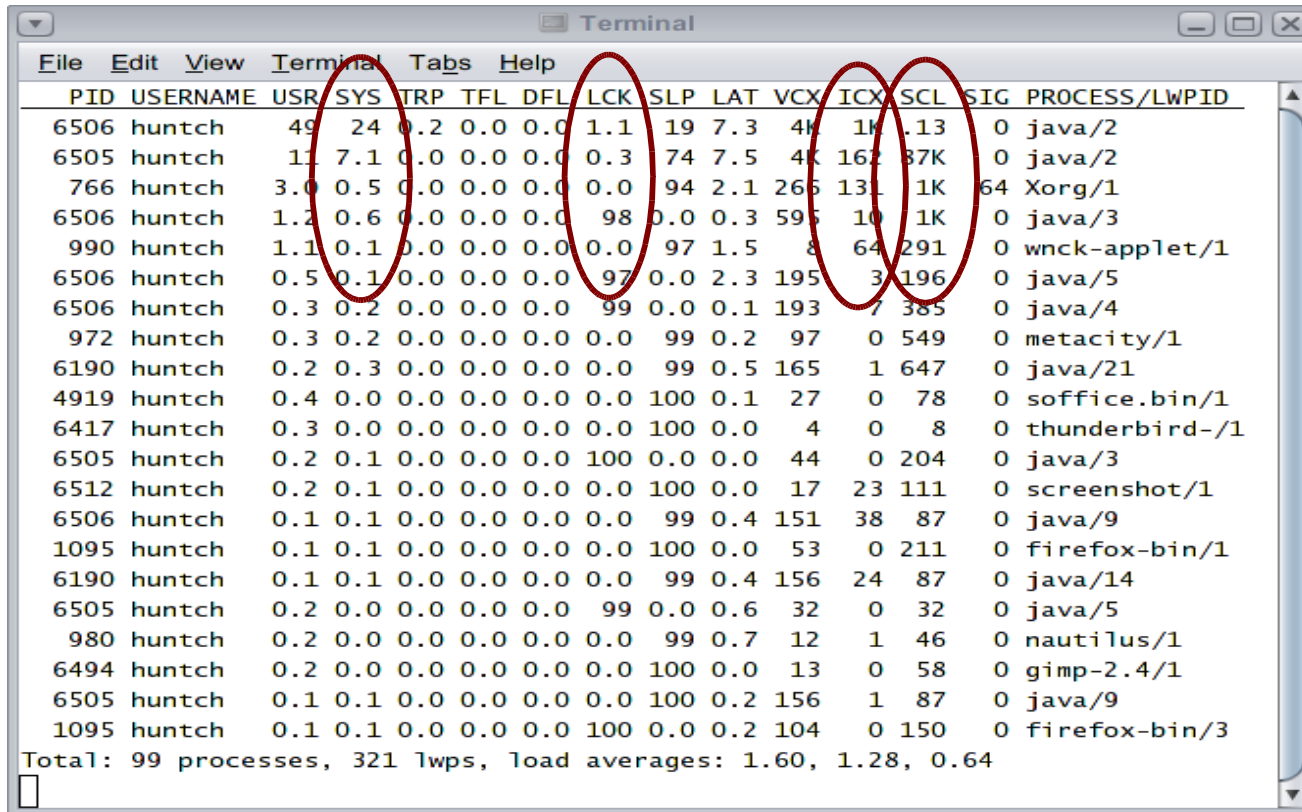
# Tools For Monitoring : mpstat

```

huntch@ditka: ~
File Edit View Terminal Tabs Help
huntch@ditka:~$ mpstat 5
CPU minf mjf xcal intr ithr csw icsw migr smtx srw syscl usr sys wt idl
0 21 1 6 319 133 261 18 23 3 0 551 4 3 0 92
1 19 1 4 36 14 59 15 23 3 0 491 4 3 0 93
2 17 1 4 19 12 61 13 23 3 0 355 4 3 0 90
3 18 1 4 16 15 49 12 23 3 0 390 4 3 0 91
CPU minf mjf xcal intr ithr csw icsw migr smtx srw syscl usr sys wt idl
0 28 2 0 192 83 92 32 14 2 0 185 78 15 0 7
1 49 1 0 37 1 80 28 16 2 0 139 80 16 0 4
2 28 1 0 20 7 94 34 17 1 0 283 83 12 0 5
3 39 1 2 52 1 99 36 16 3 0 219 74 19 0 7
CPU minf mjf xcal intr ithr csw icsw migr smtx srw syscl usr sys wt idl
0 34 0 2 171 75 78 32 12 1 0 173 90 9 0 2
1 38 1 0 39 1 84 29 13 2 0 153 66 12 0 23
2 28 8 0 21 9 97 31 20 2 0 167 67 13 0 20
3 35 3 1 43 1 98 29 20 3 0 190 52 25 0 23
  
```

- Data of interest
  - > kernel cpu utilization, locks, system calls, interrupts, migrations

# Tools For Monitoring : prstat -Lm



PID	USERNAME	USR	SYS	TRP	TFL	DFL	LCK	SLP	LAT	VCX	ICX	SCL	SIG	PROCESS/LWPID
6506	huntch	49	24	0.2	0.0	0.0	1.1	19	7.3	4K	1K	.13	0	java/2
6505	huntch	11	7.1	0.0	0.0	0.0	0.3	74	7.5	4K	162	87K	0	java/2
766	huntch	3.0	0.5	0.0	0.0	0.0	0.0	94	2.1	266	131	1K	64	Xorg/1
6506	huntch	1.2	0.6	0.0	0.0	0.0	98	0.0	0.3	595	10	1K	0	java/3
990	huntch	1.1	0.1	0.0	0.0	0.0	0.0	97	1.5	8	64	291	0	wnck-applet/1
6506	huntch	0.5	0.1	0.0	0.0	0.0	97	0.0	2.3	195	3	196	0	java/5
6506	huntch	0.3	0.2	0.0	0.0	0.0	99	0.0	0.1	193	7	385	0	java/4
972	huntch	0.3	0.2	0.0	0.0	0.0	0.0	99	0.2	97	0	549	0	metacity/1
6190	huntch	0.2	0.3	0.0	0.0	0.0	0.0	99	0.5	165	1	647	0	java/21
4919	huntch	0.4	0.0	0.0	0.0	0.0	0.0	100	0.1	27	0	78	0	soffice.bin/1
6417	huntch	0.3	0.0	0.0	0.0	0.0	0.0	100	0.0	4	0	8	0	thunderbird-/1
6505	huntch	0.2	0.1	0.0	0.0	0.0	100	0.0	0.0	44	0	204	0	java/3
6512	huntch	0.2	0.1	0.0	0.0	0.0	0.0	100	0.0	17	23	111	0	screenshot/1
6506	huntch	0.1	0.1	0.0	0.0	0.0	0.0	99	0.4	151	38	87	0	java/9
1095	huntch	0.1	0.1	0.0	0.0	0.0	0.0	100	0.0	53	0	211	0	firefox-bin/1
6190	huntch	0.1	0.1	0.0	0.0	0.0	0.0	99	0.4	156	24	87	0	java/14
6505	huntch	0.2	0.0	0.0	0.0	0.0	99	0.0	0.6	32	0	32	0	java/5
980	huntch	0.2	0.0	0.0	0.0	0.0	0.0	99	0.7	12	1	46	0	nautilus/1
6494	huntch	0.2	0.0	0.0	0.0	0.0	0.0	100	0.0	13	0	58	0	gimp-2.4/1
6505	huntch	0.1	0.1	0.0	0.0	0.0	0.0	100	0.2	156	1	87	0	java/9
1095	huntch	0.1	0.1	0.0	0.0	0.0	100	0.0	0.2	104	0	150	0	firefox-bin/3

Total: 99 processes, 321 lwps, load averages: 1.60, 1.28, 0.64

- Data of interest
  - > kernel cpu utilization, locks, system calls, involuntary context switches

# kernel : What to look for

- Why are high sys / kernel cpu, run queue depth, lock contention, migrations and context switching are important to monitor?
  - > Discussion
- What do they indicate when each is observed?
  - > Discussion
- How to do you address each of these problems?
  - > Discussion

# Monitoring Tools at the JVM-Level

# Tools For Monitoring: JVM

- Garbage Collection
  - > -verbose:gc
  - > -XX:+PrintGCTimeStamps
  - > -XX:+PrintGCDetails
  - > -XX:+PrintGCApplicationStoppedTime
  - > -XX:+PrintGCApplicationConcurrentTime
  - > jstat, jps
  - > Jconsole
  - > VisualVM
  - > VisualGC
  - > dtrace (HotSpot JDK 6 contains samples)

# Tools For Monitoring: JVM

- Garbage Collection Data of Interest
  - > Frequency and duration of collections
  - > Java heap usage
  - > Number of application threads
  - > Lock contention
  - > CPU usage

# Tools For Monitoring : `-verbose:gc`

- `-verbose:gc`
  - > [GC 1884K->1299K(5056K), 0.0031820 secs]
- With `-XX:+PrintGCTimeStamps`
  - > 3.791: [GC 1884K->1299K(5056K), 0.0031820 secs]
- Data of interest
  - > frequency and duration, heap usage
- Explain what pattern(s) indicate potential problems.
  - > Quick demo

# Tools For Monitoring : GCDetails

- **-XX:+PrintGCDetails**
- [GC [DefNew: 490K->64K(960K), 0.0032800 secs] 5470K->5151K(7884K), 0.0033270 secs] [Times: user=0.00 sys=0.00, real=0.00 secs]
- [Full GC (System) [Tenured: 5087K->5151K(6924K), 0.0971070 secs] 6047K->5151K(7884K), [Perm : 11178K->11178K(16384K)], 0.0972120 secs] [Times: user=0.10 sys=0.01, real=0.10 secs]
- **Data of interest**
  - > frequency and duration, heap usage

# Tools For Monitoring : pause time

- `-XX:+PrintGCApplicationStoppedTime`
- `-XX:+PrintGCApplicationConcurrentTime`
- Helpful when tuning pause time sensitive applications
- Useful for identifying odd pause time issues when combined with gc timestamps and gc duration.

# Tools For Monitoring : jps

- jps
  - > included in the HotSpot JDK.
  - > command line utility to find running java processes.
  - > `jps [-q] [-mlvV] [<hostid>]` where `<hostid> = <hostname>[:<port>]`

# Tools For Monitoring : jstat

- jstat
  - > included in the HotSpot JDK.
  - > command line utility.
  - > `jstat -<option> [-t] [-h<lines>] <vmid> [<internal> [<count>]]`
  - > Garbage collection option(s):
    - `-gc`, `-gccapacity`, `-gccause`, `-gcnew`, `-gcnewcapacity`, `-gcold`, `-gcoldcapacity`, `-gcpermcapacity`, `-gcutil`

# Tools For Monitoring : jstat

- *Beware:* When using the concurrent collector (CMS), jstat reports two full gc events per CMS cycle, obviously misleading.
- But, young generation stats are accurate with CMS.

# Tools For Monitoring : jconsole

- jconsole
  - > Monitoring and management GUI console.
  - > Included in the HotSpot JDK.
  - > Attach local or remote.
  - > Monitor internals of a target JVM.
  - > Monitor multiple JVMs.
  - > Explain what patterns indicate potential problems.

# Tools For Monitoring : jconsole

- Endless observability
  - > MBean support for
    - JVM memory usage by memory pool / spaces
    - Class loading, JIT compilation, garbage collector, runtime, threading and logging
    - Thread monitor contention
  - > Graphical view of heap memory, threads, cpu usage and class loading

# Tools For Monitoring : VisualVM

- VisualVM
  - > Open source project at <https://visualvm.dev.java.net>
  - > Integrates several existing JDK software tools, lightweight memory and CPU profiling capabilities.
    - JConsole
    - Subset of NetBeans Profiler
  - > Includes performance analysis and troubleshooting abilities.
    - Thread deadlock detection
    - Thread monitor contention

# Tools For Monitoring : VisualVM

- VisualVM
  - > Can be further extended with specific functionality for target application via additional plug-in or extending an existing plug-in.
    - Possibilities include; GlassFish performance monitoring plug-in, JavaDB performance monitoring plug-in and external vendors such as WebSphere performance monitoring plug-in.
  - > Plugins, enhancements and updates delivered through VisualVM plug-in center.

# Tools For Monitoring : VisualVM

- VisualVM
  - > Explain what patterns indicate potential performance issues.

# Tools For Monitoring : VisualGC

- VisualGC
  - > Standalone GUI or VisualVM plug-in.
  - > Not included in HotSpot JDK. Separate download.
  - > Visually observe garbage collection behavior. (A picture is worth a thousand words).
  - > Also includes classloading and JIT compilation information.

# Tools For Monitoring : JVM

- JIT Compilation
  - > jstat
  - > Jconsole
  - > VisualVM
  - > VisualGC
  - > -XX:+PrintCompilation (can be intrusive)
  - > -XX:+LogCompilation (can be intrusive)
  - > DTrace (HotSpot JDK 6 contains samples)
- Data of interest
  - > frequency, duration, possible opt / de-opt cycles, failed compilations

# Tools For Monitoring : JIT

- **-XX:+PrintCompilation**

- 1 java.util.Properties\$LineNumberReader::readLine (452 bytes)
- 2 java.lang.String::hashCode (60 bytes)
- 3 java.lang.String::equals (88 bytes)
- 3 made not entrant (2) java.lang.String::equals (88 bytes)
- 4 java.lang.Object::<init> (1 bytes)
- 5 java.lang.String::indexOf (151 bytes)

- **Data of interest**

- > frequency, duration, possible opt / de-opt cycles, (explain the patterns which indicate trouble)

# Tools For Monitoring : JIT

- -XX:+LogCompilation
  - > Beware, it can be intrusive
- Will probably need someone from JIT compiler team to analyze it.
- Data of interest
  - > frequency, duration, possible opt / de-opt cycles

# Tools For Monitoring : JIT

- Using .hotspot\_compiler file
- When & why to use it?
  - > JIT compiler in an endless loop attempting a “heroic” optimization which will not converge
  - > JIT compiler in a de-optimization – re-optimization cycle
  - > JIT compiler producing 'bad' code resulting in a core dump or other severe problem

# Tools For Monitoring : JIT

- Using `.hotspot_compiler` file, continued ...
- What is the format?
  - > exclude `A/B/C/D methodName` where
  - > `A.B.C.D` is the fully qualified package and class name and `methodName` is the method name.
  - > To exclude `java.util.HashMap.clear()`, specify:
    - exclude `java/util/HashMap clear`

# Monitoring Tools at the Application-Level

# Tools For Monitoring : Application

- Application throughput and / or responsiveness
  - > JConsole (application Mbeans)
  - > Extend VisualVM with a plug-in to gather Java application data of interest and monitor the application with VisualVM
  - > Application log
  - > Specialized DTrace scripts
- Data of interest
  - > critical application information and instrumentation



# Performance Monitoring (focused on Java SE)

